Panduan Pengguna

Amazon Simple Storage Service

Versi API 2006-03-01

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Simple Storage Service: Panduan Pengguna

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara pelanggan, atau dengan cara apa pun yang menghina atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan milik dari pemiliknya masing-masing, yang mungkin berafiliasi dengan, terhubung ke, atau disponsori oleh Amazon.

Table of Contents

Apa itu Amazon S3? ... 1
Fitur-fitur Amazon S3 ... 1
Kelas penyimpanan ... 1
Manajemen penyimpanan ... 2
Manajemen akses dan keamanan ... 3
Pemrosesan data ... 4
Pencatatan dan pemantauan penyimpanan ... 4
Analitik dan wawasan ... 5
Konsistensi kuat ... 5
Cara kerja Amazon S3 ... 6
Bucket ... 7
Objek ... 7
Kunci ... 8
Penentuan Versi S3 ... 8
ID Versi ... 8
Kebijakan bucket ... 8
Titik Akses S3 ... 9
Daftar kontrol akses (ACL) ... 9
Wilayah ... 10
Model konsistensi data Amazon S3 ... 10
Aplikasi yang bersamaan ... 12
Layanan terkait ... 13
Mengakses Amazon S3 ... 14
AWS Management Console ... 14
AWS Command Line Interface ... 14
AWS SDK ... 15
API REST Amazon S3 ... 15
Membayar Amazon S3 ... 16
Kepatuhan PCI DSS ... 16
Memulai ... 17
Pengaturan ... 18
Mendaftar untuk Akun AWS ... 18
Buat pengguna dengan akses administratif ... 19
Langkah 1: Buat bucket ... 20

Langkah 2: Unggah satu objek ... 27
Langkah 3: Mengunduh objek ... 28
Menggunakan konsol S3 ... 28
Langkah 4: Salin objek ... 29
Langkah 5: Hapus objek dan bucket Anda ... 30
Menghapus objek ... 31
Mengosongkan bucket Anda ... 31
Menghapus bucket Anda ... 32
Langkah selanjutnya ... 32
Memahami kasus penggunaan umum ... 33
Mengontrol akses ke bucket dan objek Anda ... 33
Mengelola dan memantau penyimpanan Anda ... 34
Mengembangkan dengan Amazon S3 ... 35
Belajar dari tutorial ... 36
Menjelajahi pelatihan dan dukungan ... 38
Tutorial ... 39
Memulai ... 36
Mengoptimalkan biaya penyimpanan ... 37
Mengelola penyimpanan ... 37
Hosting video dan situs web ... 37
Pemrosesan data ... 37
Melindungi data ... 37
Mengubah data dengan S3 Object Lambda ... 40
Prasyarat ... 42
Langkah 1: Buat ember S3 ... 44
Langkah 2: Unggah file ke bucket S3 ... 45
Langkah 3: Buat titik akses S3 ... 46
Langkah 4: Buat fungsi Lambda ... 47
Langkah 5: Konfigurasikan kebijakan IAM untuk peran eksekusi fungsi Lambda Anda ... 53
Langkah 6: Buat Titik Akses Lambda Objek S3 ... 54
Langkah 7: Lihat data yang diubah ... 55
Langkah 8: Membersihkan ... 58
Langkah selanjutnya ... 61
Mendeteksi dan menyunting data PII ... 62
Prasyarat: Buat pengguna IAM dengan izin ... 63
Langkah 1: Buat ember S3 ... 66

Langkah 2: Unggah file ke bucket S3 ... 66
Langkah 3: Buat titik akses S3 ... 67
Langkah 4: Konfigurasikan dan terapkan fungsi Lambda bawaan ... 69
Langkah 5: Buat Titik Akses Lambda Objek S3 ... 70
Langkah 6: Gunakan S3 Object Lambda Access Point untuk mengambil file yang disunting ... 71
Langkah 7: Bersihkan ... 72
Langkah selanjutnya ... 76
Hosting streaming video ... 77
Prasyarat: Daftarkan dan konfigurasikan domain khusus dengan Route 53 ... 79
Langkah 1: Buat ember S3 ... 80
Langkah 2: Unggah video ke bucket S3 ... 81
Langkah 3: Buat identitas akses CloudFront asal ... 81
Langkah 4: Buat CloudFront distribusi ... 82
Langkah 5: Akses video melalui CloudFront distribusi ... 84
Langkah 6: Konfigurasikan CloudFront distribusi Anda untuk menggunakan nama domain kustom Anda ... 85
Langkah 7: Akses video S3 melalui CloudFront distribusi dengan nama domain khusus ... 90
(Opsional) Langkah 8: Lihat data tentang permintaan yang diterima oleh CloudFront distribusi Anda ... 91
Langkah 9: Membersihkan ... 91
Langkah selanjutnya ... 96
Batch-transcoding video ... 97
Prasyarat ... 99
Langkah 1: Buat bucket S3 untuk file media keluaran ... 99
Langkah 2: Buat peran IAM untuk MediaConvert ... 101
Langkah 3: Buat peran IAM untuk fungsi Lambda Anda ... 102
Langkah 4: Buat fungsi Lambda untuk transcoding video ... 104
Langkah 5: Konfigurasikan Inventaris Amazon S3 untuk bucket sumber S3 Anda ... 122
Langkah 6: Buat peran IAM untuk Operasi Batch S3 ... 126
Langkah 7: Buat dan jalankan pekerjaan Operasi Batch S3 ... 129
Langkah 8: Memeriksa file media output dari bucket tujuan S3 Anda ... 134
Langkah 9: Membersihkan ... 135
Langkah selanjutnya ... 138
Mengonfigurasi situs web statis ... 138
Langkah 1: Buat bucket ... 139

Langkah 2: Mengaktifkan hosting situs web statis ... 140
Langkah 3: Mengedit pengaturan Blokir Akses Publik ... 141
Langkah 4: Menambahkan kebijakan bucket yang membuat konten bucket Anda tersedia untuk umum ... 143
Langkah 5: Mengonfigurasi dokumen indeks ... 145
Langkah 6: Mengonfigurasi dokumen kesalahan ... 146
Langkah 7: Menguji titik akhir situs web Anda ... 147
Langkah 8: Membersihkan ... 148
Mengonfigurasi situs web statis menggunakan domain kustom ... 148
Sebelum Anda mulai ... 150
Langkah 1: Mendaftarkan domain kustom dengan Route 53 ... 150
Langkah 2: Membuat dua bucket ... 150
Langkah 3: Mengonfigurasi bucket Domain root ... 151
Langkah 4: Mengonfigurasi bucket subdomain untuk pengalihan ... 153
Langkah 5: Mengonfigurasi pencatatan ... 153
Langkah 6: Mengunggah konten indeks dan situs web ... 155
Langkah 7: Mengunggah dokumen kesalahan ... 156
Langkah 8: Mengedit pengaturan Blokir Akses Publik ... 157
Langkah 9: Melampirkan kebijakan bucket ... 159
Langkah 10: Menguji titik akhir domain Anda ... 160
Langkah 11: Menambahkan catatan alias ... 161
Langkah 12: Menguji situs webnya ... 166
Mempercepat situs web Anda dengan Amazon CloudFront ... 167
Membersihkan contoh sumber daya ... 171
Bekerja dengan bucket ... 174
Gambaran umum bucket ... 175
Tentang izin ... 176
Mengelola akses publik ke bucket ... 177
Konfigurasi bucket ... 178
Peraturan penamaan ... 182
Aturan penamaan bucket tujuan umum ... 182
Aturan penamaan bucket direktori ... 184
Mengakses dan mendaftarkan bucket ... 184
... 184
Mendaftarkan bucket ... 187
Membuat bucket ... 188

Melihat properti bucket ... 200
Mengosongkan bucket ... 203
Mengosongkan ember dengan dikonfigurasi AWS CloudTrail ... 206
Menghapus bucket ... 206
Mengatur enkripsi bucket default ... 212
Menggunakan enkripsi SSE-KMS untuk operasi lintas akun ... 214
Menggunakan enkripsi default menggunakan replikasi ... 214
Menggunakan Kunci Bucket Amazon S3 dengan enkripsi default ... 215
Mengonfigurasi enkripsi default ... 215
Memantau enkripsi default ... 221
Mountpoint untuk Amazon S3 ... 222
Instalasi Mountpoint ... 223
Mengkonfigurasi dan menggunakan Mountpoint ... 228
Mengonfigurasi Transfer Acceleration ... 231
Mengapa menggunakan Percepatan Transfer? ... 232
Persyaratan untuk menggunakan Transfer Acceleration ... 232
Memulai ... 234
Mengaktifkan Transfer Acceleration ... 236
Alat Perbandingan Kecepatan ... 243
Menggunakan Pembayaran Pemohon ... 244
Cara kerja Pembayaran Pemohon ... 245
Mengonfigurasi Pembayaran Pemohon ... 246
Mengambil konfigurasi requestPayment ... 248
Mengunduh objek dari ember Requester Pays ... 249
Pembatasan dan batasan ... 250
Bekerja dengan objek ... 252
Objek ... 253
Sub-sumber daya ... 254
Membuat kunci objek ... 255
Panduan penamaan kunci objek ... 256
Bekerja dengan metadata ... 260
Metadata objek yang ditentukan sistem ... 260
Metadata objek yang ditentukan pengguna ... 264
Mengedit metadata objek ... 266
Mengunggah Objek ... 268
Menggunakan unggahan multibagian ... 282

Proses pengunggahan multibagian ... 284
Checksum dengan operasi unggahan multibagian ... 286
Operasi pengunggahan multibagian serentak ... 287
Unggahan multibagian dan harga ... 287
Dukungan API untuk unggahan multibagian ... 288
AWS Command Line Interface dukungan untuk upload multipart ... 289
AWS Dukungan SDK untuk unggahan multipart ... 289
API dan izin unggahan multibagian ... 290
Mengonfigurasi konfigurasi siklus hidup ... 293
Pengunggahan objek menggunakan unggahan multibagian ... 297
Mengunggah sebuah direktori ... 322
Mendaftarkan unggahan multibagian ... 325
Melacak unggahan multibagian ... 328
Membatalkan unggahan multibagian ... 331
Menyalin objek ... 337
Batas unggahan multibagian ... 344
Menyalin, memindahkan, dan mengganti nama objek ... 344
Untuk menyalin objek ... 347
Untuk memindahkan objek ... 358
Untuk mengganti nama objek ... 359
Mengunggah objek ... 360
Mengunduh objek ... 361
Mengunduh beberapa objek ... 363
Mengunduh bagian dari suatu objek ... 365
Mengunduh objek dari Akun AWS yang lain ... 366
Mengunduh objek yang diarsipkan ... 367
Memecahkan masalah pengunduhan objek ... 367
Memeriksa integritas objek ... 367
Menggunakan algoritma checksum yang didukung ... 368
Menggunakan Content-MD5 saat mengunggah objek ... 377
Menggunakan Content-MD5 dan ETag untuk memverifikasi objek yang diunggah ... 377
Menggunakan checksum trailing ... 378
Menggunakan checksum tingkat bagian untuk unggahan multibagian ... 379
Menghapus objek ... 380
Secara terprogram menghapus objek dari sebuah bucket yang diaktifkan dengan versi ... 381
Menghapus objek dari bucket yang mengaktifkan MFA ... 382

Menghapus satu objek ... 382
Menghapus beberapa objek ... 394
Mengatur dan membuat daftar objek ... 397
Menggunakan prefiks ... 398
Membuat daftar objek ... 400
Menggunakan folder ... 403
Melihat gambaran umum objek ... 407
Melihat properti objek ... 408
Bekerja dengan URL yang telah ditandatangani ... 410
Siapa yang dapat membuat URL yang telah ditandatangani ... 411
Waktu kedaluwarsa untuk URL yang telah ditandatangani ... 411
Pembatasan kemampuan URL yang telah ditandatangani ... 412
Berbagi objek dengan URL yang telah ditandatangani ... 414
Mengunggah objek dengan URL yang telah ditandatangani sebelumnya ... 417
Mengubah objek ... 419
Membuat Titik Akses Objek Lambda ... 421
Menggunakan Titik Akses Lambda Objek Amazon S3 ... 436
Pertimbangan keamanan ... 440
Menulis fungsi Lambda ... 447
Menggunakan fungsi AWS yang dibangun ... 479
Praktik terbaik dan pedoman untuk S3 Lambda Objek ... 481
Tutorial S3 Lambda Objek ... 483
Men-debug S3 Lambda Objek ... 483
Apa itu S3 Express One Zone? ... 485
Gambaran Umum ... 487
Zona Ketersediaan Tunggal ... 487
Bucket direktori ... 487
Titik akhir dan titik akhir VPC gateway ... 488
Otorisasi berbasis sesi ... 488
Fitur S3 Express One Zone ... 488
Manajemen akses dan keamanan ... 489
Pencatatan dan pemantauan ... 489
Manajemen objek ... 490
AWS SDK dan pustaka klien ... 491
Enkripsi dan perlindungan data ... 491
AWS Versi Tanda Tangan 4 (SigV4) ... 492

Konsistensi kuat ... 492
Layanan terkait ... 492
Langkah selanjutnya ... 493
Apa yang membuat S3 Express One Zone berbeda? ... 494
Perbedaan S3 Express One Zone ... 494
Operasi API yang didukung oleh S3 Express One Zone ... 496
Fitur Amazon S3 tidak didukung oleh S3 Express One Zone ... 497
Memulai dengan S3 Express One Zone ... 498
Siapkan AWS Identity and Access Management (IAM) dengan S3 Express One Zone ... 498
Konfigurasikan titik akhir VPC gateway ... 499
Bekerja dengan S3 Express One Zone dengan menggunakan konsol S3, AWS CLI, dan SDK AWS ... 499
Jaringan untuk S3 Express One Zone ... 501
Titik akhir ... 501
Mengkonfigurasi titik akhir gateway VPC ... 502
Ember direktori ... 503
Zona Ketersediaan ... 504
Nama bucket direktori ... 505
Direktori ... 505
Nama kunci ... 506
Manajemen akses ... 506
Bekerja dengan bucket direktori ... 506
Aturan penamaan bucket direktori ... 506
Membuat bucket direktori ... 507
Melihat properti ... 517
Mengelola kebijakan bucket ... 517
Mengosongkan bucket direktori ... 522
Menghapus bucket direktori ... 524
Mendaftar bucket direktori ... 526
`HeadBucket` contoh ... 529
Bekerja dengan objek di dalam bucket direktori ... 530
Mengimpor objek ke dalam bucket direktori ... 531
Menggunakan Operasi Batch dengan S3 Express One Zone ... 533
Mengunggah Objek ... 535
Menggunakan unggahan multibagian dengan bucket direktori ... 538
Menyalin objek ... 567

Menghapus objek ... 572
Mengunduh objek ... 576
`HeadObject` contoh ... 578
Keamanan untuk S3 Express One Zone ... 579
Perlindungan data dan enkripsi ... 580
IAM untuk S3 Express One Zone ... 581
Kebijakan berbasis identitas ... 597
Kebijakan bucket ... 598
`CreateSession`otorisasi ... 601
Praktik terbaik keamanan ... 602
Mengoptimalkan kinerja S3 Express One Zone ... 605
Pedoman kinerja dan pola desain ... 606
Mengembangkan dengan S3 Express One Zone ... 610
Zona Ketersediaan dan Wilayah S3 Express One Zone ... 611
Titik akhir Regional dan Zona ... 613
Operasi S3 Express One Zone API ... 613
Bekerja dengan titik akses ... 615
Mengonfigurasi kebijakan IAM ... 616
Contoh kebijakan titik akses ... 616
Kunci syarat ... 621
Mendelegasikan kontrol akses ke titik akses ... 622
Memberikan izin untuk titik akses lintas akun ... 623
Membuat titik akses ... 623
Aturan penamaan titik akses Amazon S3 ... 624
Membuat titik akses ... 624
Membuat titik akses terbatas pada VPC ... 627
Mengelola akses publik ... 630
Menggunakan titik akses ... 631
Mengakses bucket melalui titik akses S3 ... 632
Pemantauan dan pencatatan ... 632
Mengelola titik akses ... 634
Menggunakan alias gaya bucket untuk titik akses Anda ... 637
Menggunakan titik akses dengan operasi Amazon S3 ... 639
Pembatasan dan batasan ... 643
Bekerja dengan Titik Akses Multi-Wilayah ... 645
Membuat Titik Akses Multi-Wilayah ... 646

Aturan penamaan Titik Akses Multi-Wilayah Amazon S3 ... 648
Aturan dalam memilih bucket untuk Titik Akses Multi-Wilayah Amazon S3 ... 649
Membuat Titik Akses Multi-Wilayah Amazon S3 ... 650
Memblokir akses publik dengan Titik Akses Multi-Wilayah Amazon S3 ... 653
Melihat detail konfigurasi Titik Akses Multi-Wilayah Amazon S3 ... 653
Menghapus Titik Akses Multi-Wilayah ... 655
Mengonfigurasi Titik Akses Multi-Wilayah ... 656
Mengonfigurasi AWS PrivateLink ... 656
Menghapus akses ke Titik Akses Multi-Wilayah dari titik akhir VPC ... 659
Menggunakan Titik Akses Multi Wilayah ... 660
Nama host Titik Akses Multi-Wilayah ... 661
Titik akses Multi-wilayah dan Amazon S3 Transfer Acceleration ... 662
Izin ... 663
Pembatasan dan batasan ... 671
Meminta perutean ... 674
Konfigurasi failover ... 675
Replikasi ember ... 683
Operasi API yang didukung ... 692
Pemantauan dan logging ... 708
Keamanan ... 712
Perlindungan data ... 713
Enkripsi data ... 715
enkripsi di sisi server ... 716
Menggunakan enkripsi di sisi klien ... 807
Privasi antar jaringan ... 808
Lalu lintas antara layanan dan aplikasi serta klien on-premise ... 808
Lalu lintas antar AWS sumber daya di Wilayah yang sama ... 808
AWS PrivateLink untuk Amazon S3 ... 809
Jenis titik akhir VPC ... 809
Pembatasan dan batasan AWS PrivateLink untuk Amazon S3 ... 810
Membuat titik akhir VPC ... 811
Mengakses titik akhir antarmuka Amazon S3 ... 811
DNS privat ... 811
Mengakses bucket, titik akses, serta operasi API Amazon S3 Control dari titik akhir antarmuka S3 ... 814
Memperbarui konfigurasi DNS on-premise ... 820

Membuat kebijakan titik akhir VPC ... 822
Manajemen Akses ... 826
Sumber daya S3 ... 827
Identitas ... 832
Alat manajemen akses ... 834
Tindakan ... 840
Kasus penggunaan manajemen akses ... 841
Pemecahan masalah manajemen akses ... 848
Identity and Access Management ... 850
Mengelola akses dengan S3 Access Grants ... 1028
Mengelola Akses dengan ACL ... 1110
Memblokir akses publik ... 1153
Meninjau akses bucket ... 1171
Memverifikasi pemilik bucket ... 1178
Mengontrol kepemilikan objek ... 1183
Menggunakan CORS ... 1225
Berbagi sumber daya lintas asal: Skenario kasus penggunaan ... 1226
Bagaimana Amazon S3 mengevaluasi konfigurasi CORS pada bucket? ... 1226
Bagaimana Titik Akses Lambda Objek mendukung CORS ... 1227
Konfigurasi CORS ... 1227
Mengonfigurasi CORS ... 1233
Pencatatan dan pemantauan ... 1243
Validasi Kepatuhan ... 1246
Ketangguhan ... 1248
Enkripsi cadangan ... 1250
Keamanan infrastruktur ... 1251
Konfigurasi dan analisis kerentanan ... 1252
Praktik terbaik keamanan ... 1253
Praktik terbaik keamanan Amazon S3 ... 1253
Praktik Terbaik Pemantauan dan Audit Amazon S3 ... 1259
Memantau keamanan data ... 1264
Mengelola penyimpanan ... 1268
Menggunakan Penentuan Versi S3 ... 1269
Bucket tanpa versi, dengan dukungan Penentuan Versi, dan dengan Penentuan Versi ditangguhkan ... 1269
Menggunakan Penentuan Versi S3 dengan Siklus Hidup S3 ... 1270

Penentuan Versi S3 ... 1271
Mengaktifkan Penentuan Versi pada bucket ... 1275
Mengonfigurasi penghapusan MFA ... 1283
Bekerja dengan objek dengan Penentuan Versi yang diaktifkan ... 1285
Bekerja dengan objek dengan Penentuan Versi ditangguhkan ... 1315
MenggunakanAWS Backup untuk Amazon S3 ... 1320
Bekerja dengan objek yang diarsipkan ... 1321
Memulihkan objek dari S3 Glacier ... 1322
Memulihkan objek dari S3 Intelligent-Tiering ... 1322
Menggunakan Operasi Batch S3 dengan permintaan pemulihan ... 1322
Waktu pemulihan ... 1323
Opsi pengambilan arsip ... 1323
Memulihkan objek yang diarsipkan ... 1326
Menggunakan Kunci Objek ... 1334
Cara kerja Kunci Objek S3 ... 1335
Pertimbangan Kunci Objek ... 1339
Mengonfigurasi Kunci Objek ... 1345
Mengelola kelas penyimpanan ... 1355
Objek yang sering diakses ... 1356
Secara otomatis mengoptimalkan data dengan pola akses yang berubah atau tidak diketahui ... 1357
Objek yang jarang diakses ... 1359
Objek yang jarang diakses ... 1360
Amazon S3 di Outposts ... 1361
Membandingkan kelas penyimpanan ... 1362
Mengatur kelas penyimpanan objek ... 1363
Kelas penyimpanan Amazon S3 Glacier ... 1365
Membandingkan kelas penyimpanan S3 Glacier ... 1365
S3 Glacier Instant Retrieval ... 1366
S3 Glacier Flexible Retrieval ... 1366
S3 Glacier Deep Archive ... 1367
Penyimpanan arsip ... 1368
Bagaimana kelas penyimpanan ini berbeda dari layanan S3 Glacier ... 1369
Amazon S3 Intelligent-Tiering ... 1369
Cara kerja S3 Intelligent-Tiering ... 1370
Menggunakan S3 Intelligent-Tiering ... 1373

Mengelola S3 Intelligent-Tiering ... 1378
Mengelola siklus hidup ... 1382
Mengelola siklus hidup objek ... 1383
Membuat konfigurasi siklus hidup ... 1384
Transisi objek ... 1385
Mengakhiri objek ... 1394
Menetapkan konfigurasi siklus hidup ... 1397
Menggunakan konfigurasi bucket lainnya ... 1416
Mengonfigurasi notifikasi peristiwa Siklus Hidup ... 1419
Elemen konfigurasi Siklus Hidup ... 1421
Contoh konfigurasi Siklus Hidup S3 ... 1433
Mengelola inventaris ... 1451
Bucket inventaris Amazon S3 ... 1453
Daftar inventaris ... 1453
Mengonfigurasi Inventaris Amazon S3 ... 1457
Menyiapkan notifikasi peristiwa untuk penyelesaian inventaris ... 1466
Menemukan inventaris ... 1467
Membuat kueri inventaris dengan Athena ... 1472
Mengkonversi string ID versi kosong ke string null ... 1478
Menggunakan bidang ACL Objek ... 1481
Mereplikasi objek ... 1483
Mengapa menggunakan replikasi? ... 1484
Kapan menggunakan Replikasi Lintas-Wilayah ... 1485
Kapan menggunakan Replikasi Lintas-Wilayah ... 1486
Kapan menggunakan replikasi dua arah (replikasi bidireksional) ... 1486
Kapan menggunakan Replikasi Batch S3 ... 1487
Persyaratan beban kerja dan replikasi langsung ... 1487
Apa yang direplikasi? ... 1488
Persyaratan dan pertimbangan untuk replikasi ... 1492
Menyiapkan replikasi langsung ... 1496
Mengelola atau menjeda replikasi langsung ... 1585
Memantau kemajuan dan mendapatkan status ... 1586
Mereplikasi objek yang ada ... 1600
Menggunakan tag objek ... 1613
Operasi API terkait pemberian tag objek ... 1616
Konfigurasi tambahan ... 1617

Kontrol akses ... 1618
Mengelola tag objek ... 1621
Menggunakan tag alokasi biaya ... 1627
Info Selengkapnya ... 1628
Pelaporan penagihan dan penggunaan ... 1629
Laporan penagihan ... 1630
Laporan penggunaan ... 1633
Memahami laporan penagihan dan penggunaan ... 1636
Penagihan untuk respons kesalahan Amazon S3 ... 1662
Menggunakan Amazon S3 Select ... 1674
Persyaratan dan batasan ... 1675
Membuat permintaan ... 1676
Kesalahan ... 1677
Contoh S3 Select ... 1678
Referensi SQL ... 1682
Menggunakan Operasi Batch ... 1721
Dasar-dasar Operasi Batch ... 1722
Tutorial Operasi Batch S3 ... 1723
Memberikan izin ... 1723
Membuat pekerjaan ... 1734
Operasi yang didukung ... 1757
Mengelola tugas ... 1799
Melacak status tugas dan laporan penyelesaian ... 1803
Menggunakan tanda ... 1819
Mengelola Kunci Objek S3 ... 1835
S3 Batch Operations ... 1859
Pemantauan Amazon S3 ... 1860
Alat pemantauan ... 1861
Alat otomatis ... 1861
Alat manual ... 1861
Opsi pencatatan ... 1862
Logging dengan CloudTrail ... 1865
Menggunakan CloudTrail log dengan log akses server Amazon S3 dan Log CloudWatch .. 1867
CloudTrail melacak dengan panggilan API SOAP Amazon S3 ... 1868
CloudTrail acara ... 1869
Contoh file log ... 1880

Mengaktifkan CloudTrail ... 1886
Mengidentifikasi permintaan S3 ... 1889
Pencatatan akses server ... 1897
Bagaimana cara mengaktifkan log pengiriman? ... 1897
Format kunci objek log ... 1900
Bagaimana log dikirimkan? ... 1900
Pengiriman log server dengan upaya terbaik ... 1902
Perubahan status pencatatan log bucket memerlukan waktu ... 1902
Mengaktifkan pencatatan akses server ... 1903
Format log ... 1924
Menghapus file log ... 1939
Mengidentifikasi permintaan S3 ... 1940
Memantau metrik dengan CloudWatch ... 1947
Metrik dan dimensi ... 1949
Mengakses metrik CloudWatch ... 1967
CloudWatch konfigurasi metrik ... 1968
Notifikasi Peristiwa Amazon S3 ... 1977
Ikhtisar ... 1977
Jenis dan tujuan notifikasi peristiwa ... 1979
Menggunakan SQS, SNS, dan Lambda ... 1987
Menggunakan EventBridge ... 2016
Menggunakan analitik dan wawasan ... 2027
Analisis Kelas Penyimpanan ... 2027
Cara menyiapkan analisis kelas penyimpanan ... 2028
Analisis kelas penyimpanan ... 2029
Bagaimana cara mengekspor data analisis kelas penyimpanan? ... 2031
Mengonfigurasi analisis kelas penyimpanan ... 2032
S3 Storage Lens ... 2035
Metrik dan fitur Lensa Penyimpanan S3 ... 2036
Memahami S3 Storage Lens ... 2038
Bekerja dengan Organizations ... 2049
Izin Lensa Penyimpanan S3 ... 2053
Melihat metrik penyimpanan ... 2057
Amazon S3 ... 2089
Glosarium Metrik ... 2114
Bekerja dengan S3 Storage Lens ... 2148

Bekerja dengan grup Lensa Penyimpanan ... 2196
Melacak permintaan menggunakan X-Ray ... 2237
Cara X-Ray bekerja dengan Amazon S3 ... 2238
Wilayah yang Tersedia ... 2239
Hosting situs web statis ... 2240
Titik akhir situs web ... 2241
Contoh titik akhir situs web ... 2242
Menambahkan CNAME DNS ... 2243
Menggunakan domain khusus dengan Route 53 ... 2243
Perbedaan utama antara titik akhir situs web dan titik akhir API REST ... 2243
Mengaktifkan hosting situs web ... 2244
Mengonfigurasi dokumen indeks ... 2250
Indeks dokumen dan folder ... 2250
Mengonfigurasi dokumen indeks ... 2251
Mengonfigurasi dokumen kesalahan khusus ... 2253
Kode respons HTTP Amazon S3 ... 2253
Mengonfigurasi dokumen kesalahan khusus ... 2256
Mengatur izin untuk akses situs web ... 2257
Langkat 1: Edit pengaturan Blokir Akses Publik S3 ... 2258
Langkah 2: Menambahkan kebijakan bucket ... 2260
Daftar kontrol akses objek ... 2262
Mencatat lalu lintas web ... 2263
Mengonfigurasi pengalihan ... 2264
Permintaan pengalihan ke host lain ... 2264
Konfigurasikan aturan pengalihan ... 2265
Mengalihkan permintaan untuk objek ... 2273
Mengembangkan dengan Amazon S3 ... 2276
Membuat permintaan ... 2276
Tentang kunci akses ... 2277
Meminta titik akhir ... 2279
Membuat permintaan melalui IPv6 ... 2279
Membuat permintaan menggunakan AWS SDK ... 2290
Membuat permintaan menggunakan API REST ... 2332
Menggunakan AWS CLI ... 2347
Menggunakan AWS SDK ... 2349
Bekerja dengan AWS SDK ... 2349

Antarmuka Pemrograman SDK ... 2350
Menentukan Versi Tanda Tangan di Autentikasi Permintaan ... 2351
Penggunaan REST API ... 2361
Meminta perutean ... 2362
Penanganan kesalahan ... 2369
Respons kesalahan REST ... 2369
Respons kesalahan SOAP ... 2371
Praktik terbaik kesalahan Amazon S3 ... 2372
Referensi ... 2373
Lampiran a: Menggunakan SOAP API ... 2374
Lampiran b: Mengautentikasi permintaan (versi AWS tanda tangan 2) ... 2378
Mengoptimalkan Performa Amazon S3 ... 2425
Pedoman Kinerja ... 2426
Mengukur Performa ... 2427
Menskalakan secara Horizontal ... 2428
Menggunakan Byte-Range Fetches ... 2428
Permintaan Coba Kembali ... 2428
Menggabungkan Amazon S3 dan Amazon EC2 di Wilayah yang Sama ... 2428
Gunakan Transfer Acceleration untuk Meminimalkan Latensi ... 2429
Gunakan SDK AWS terbaru ... 2429
Pola Desain Performa ... 2430
Meng-caching Konten yang Sering Diakses ... 2430
Batas Waktu dan Pengulangan untuk App yang Sensitif-Latensi ... 2431
Penskalaan Horizontal dan Paralelisasi Permintaan ... 2432
Mempercepat Transfer Data yang Berbeda Secara Geografis ... 2433
Apa itu S3 di Outposts? ... 2435
Cara kerja S3 di Outposts ... 2435
Wilayah ... 2436
Bucket ... 2436
Objek ... 2437
Kunci ... 2437
Penentuan Versi S3 ... 2438
ID Versi ... 2438
Kelas penyimpanan dan enkripsi ... 2438
Kebijakan bucket ... 2438
Titik akses S3 di Outposts ... 2439

Fitur S3 di Outposts 2440
Manajemen akses 2440
Pencatatan dan pemantauan penyimpanan 2440
Konsistensi kuat 2441
Layanan terkait 2441
Mengakses S3 di Outposts 2442
AWS Management Console 2442
AWS Command Line Interface 2442
AWS SDK 2442
Membayar untuk S3 di Outposts 2443
Langkah selanjutnya 2443
Menyiapkan Anda 2443
Pesan Pos Terdepan baru 2444
Bagaimana S3 di Outposts berbeda 2444
Spesifikasi 2444
Operasi API yang didukung 2445
Fitur Amazon S3 yang tidak didukung 2445
Pembatasan jaringan 2446
Memulai dengan S3 di Outposts 2447
Mengatur IAM 2447
Menggunakan konsol S3 2455
Menggunakan AWS CLI dan SDK for Java 2459
Jaringan untuk S3 di Outposts 2463
Memilih jenis akses jaringan Anda 2464
Mengakses S3 Anda di ember dan objek Outposts 2464
Mengelola koneksi menggunakan antarmuka jaringan elastis lintas akun 2465
Bekerja dengan S3 di bucket Outposts 2465
Bucket 2465
Titik Akses 2466
Titik akhir 2466
operasi API di S3 di S3 di Outposts 2466
Membuat dan mengelola S3 di bucket Outposts 2468
Membuat bucket 2469
Menambahkan tanda 2473
Menggunakan kebijakan bucket 2474
Buat Daftar Bucket 2483

Mendapatkan ember ... 2484
Menghapus bucket Anda ... 2486
Bekerja dengan titik akses ... 2487
Bekerja dengan ... 2500
Bekerja dengan S3 di objek Outposts ... 2507
Mengunggah objek ... 2508
Penyalinan objek ... 2510
Pengambilamakan objek ... 2512
Daftar objek ... 2515
Penghapusan objek ... 2518
Menggunakan HeadBucket ... 2523
Melakukan unggahan multipart ... 2525
Menggunakan URLs yang telah ditandatangani ... 2532
Amazon S3 di Outposts dengan Amazon EMR lokal ... 2546
Caching otorisasi dan otentikasi ... 2553
Keamanan ... 2554
Enkripsi data ... 2555
AWS PrivateLink untuk S3 di Outposts ... 2556
Kunci kebijakan Signature Version 4 (SiGv4) ... 2562
Kebijakan yang dikelola AWS ... 2566
Menggunakan peran terkait layanan ... 2567
Mengelola penyimpanan S3 di Outposts ... 2572
Mengelola versioning ... 2573
Membuat siklus aktifnya ... 2575
Replikasi objek untuk S3 di Outposts ... 2583
Berbagi S3 di Outposts ... 2614
Layanan lainnya ... 2618
Memantau S3 di Outposts ... 2619
CloudWatch metrik ... 2620
CloudWatch Acara Amazon ... 2622
CloudTrail log ... 2623
Mengembangkan dengan S3 di Outposts ... 2627
S3 di API Outposts ... 2627
Mengkonfigurasi klien kontrol S3 ... 2630
Membuat permintaan melalui IPv6 ... 2630
Contoh kode ... 2642

Tindakan 2654
AbortMultipartUpload 2657
AbortMultipartUploads 2658
CompleteMultipartUpload 2660
CopyObject 2662
CreateBucket 2681
CreateMultiRegionAccessPoint 2703
CreateMultipartUpload 2706
DeleteBucket 2707
DeleteBucketAnalyticsConfiguration 2718
DeleteBucketCors 2719
DeleteBucketEncryption 2722
DeleteBucketInventoryConfiguration 2723
DeleteBucketLifecycle 2724
DeleteBucketMetricsConfiguration 2727
DeleteBucketPolicy 2728
DeleteBucketReplication 2734
DeleteBucketTagging 2735
DeleteBucketWebsite 2736
DeleteObject 2741
DeleteObjectTagging 2758
DeleteObjects 2760
DeletePublicAccessBlock 2789
GetBucketAccelerateConfiguration 2790
GetBucketAcl 2792
GetBucketAnalyticsConfiguration 2801
GetBucketCors 2802
GetBucketEncryption 2807
GetBucketInventoryConfiguration 2808
GetBucketLifecycleConfiguration 2810
GetBucketLocation 2813
GetBucketLogging 2815
GetBucketMetricsConfiguration 2817
GetBucketNotification 2818
GetBucketPolicy 2819
GetBucketPolicyStatus 2827

GetBucketReplication 2828
GetBucketRequestPayment 2829
GetBucketTagging 2830
GetBucketVersioning 2831
GetBucketWebsite 2832
GetObject 2836
GetObjectAcl 2862
GetObjectLegalHold 2868
GetObjectLockConfiguration 2872
GetObjectRetention 2878
GetObjectTagging 2884
GetPublicAccessBlock 2886
HeadBucket 2888
HeadObject 2891
ListBucketAnalyticsConfigurations 2896
ListBucketInventoryConfigurations 2898
ListBuckets 2899
ListMultipartUploads 2910
ListObjectVersions 2914
ListObjects 2920
ListObjectsV2 2921
PutBucketAccelerateConfiguration 2941
PutBucketAcl 2944
PutBucketCors 2954
PutBucketEncryption 2963
PutBucketLifecycleConfiguration 2964
PutBucketLogging 2973
PutBucketNotification 2980
PutBucketNotificationConfiguration 2983
PutBucketPolicy 2990
PutBucketReplication 2998
PutBucketRequestPayment 3002
PutBucketTagging 3004
PutBucketVersioning 3005
PutBucketWebsite 3006
PutObject 3014

PutObjectAcl ... 3043
PutObjectLegalHold ... 3048
PutObjectLockConfiguration ... 3053
PutObjectRetention ... 3064
RestoreObject ... 3070
SelectObjectContent ... 3076
UploadPart ... 3081
Skenario ... 3083
Membuat URL yang telah ditetapkan sebelumnya ... 3084
Membuat halaman web yang mencantumkan objek Amazon S3 ... 3123
Hapus unggahan multipart yang tidak lengkap ... 3125
Mengunduh objek ke direktori lokal ... 3128
Dapatkan objek dari Titik Akses Multi-Region ... 3130
Mendapatkan objek dari bucket jika telah diubah ... 3131
Memulai bucket dan objek ... 3136
Memulai enkripsi ... 3215
Memulai dengan tanda ... 3221
Dapatkan konfigurasi penahanan hukum suatu objek ... 3224
Kunci objek Amazon S3 ... 3228
Mengelola daftar kontrol akses (ACL) ... 3314
Mengelola objek berversi dalam batch dengan fungsi Lambda ... 3319
Mengurai URI ... 3320
Melakukan penyalinan multibagian ... 3323
Melakukan pengunggahan multibagian ... 3326
Lacak unggahan dan unduhan ... 3330
Pengujian unit dan integrasi dengan SDK ... 3333
Mengunggah direktori ke bucket ... 3342
Mengunggah atau mengunduh file besar ... 3343
Mengunggah aliran ukuran yang tidak diketahui ... 3384
Gunakan checksum ... 3387
Bekerja dengan objek berversi ... 3392
Contoh nirserver ... 3399
Menginvokasi fungsi Lambda dari pemicu Amazon S3 ... 3399
Contoh lintas layanan ... 3411
Membangun aplikasi Amazon Transcribe ... 3412
Mengonversi teks menjadi ucapan dan kembali ke teks ... 3412

Membuat aplikasi nirserver untuk mengelola foto ... 3413
Membuat aplikasi penjelajah Amazon Textract ... 3418
Mendeteksi APD dalam gambar ... 3419
Mendeteksi entitas dalam teks yang diekstrak dari gambar ... 3421
Mendeteksi wajah dalam gambar ... 3421
Mendeteksi objek dalam gambar ... 3422
Mendeteksi orang dan objek dalam video ... 3426
Menyimpan EXIF dan informasi gambar lainnya ... 3427
Mengubah data dengan S3 Object Lambda ... 3428
Pemecahan Masalah ... 3429
Pecahkan masalah kesalahan Akses Ditolak (403 Forbidden) ... 3429
Kebijakan bucket dan kebijakan IAM ... 3430
Pengaturan Amazon S3 ACL ... 3433
Pengaturan S3 Block Public Access ... 3436
Pengaturan enkripsi Amazon S3 ... 3437
Pengaturan Kunci Objek S3 ... 3439
Kebijakan titik akhir VPC ... 3440
AWS Organizations kebijakan ... 3440
Pengaturan titik akses ... 3440
Memecahkan Masalah Operasi Batch ... 3441
Laporan pekerjaan tidak terkirim saat ada masalah izin atau mode retensi diaktifkan ... 3441
Kegagalan Replikasi Batch: Pembuatan manifes tidak menemukan kunci yang cocok dengan kriteria filter ... 3442
Kegagalan Replikasi Batch setelah menambahkan aturan replikasi baru ... 3442
Operasi Batch S3 gagal objek dengan kesalahan 400 InvalidRequest ... 3443
Buat kegagalan pekerjaan dengan pemberian tag pekerjaan diaktifkan ... 3443
Akses Ditolak untuk membaca manifes ... 3443
Penyelesaian masalah CORS ... 3444
Memecahkan masalah siklus hidup ... 3445
Saya menjalankan operasi daftar di bucket saya dan melihat objek yang menurut saya sudah kedaluwarsa atau ditransisikan oleh aturan siklus hidup. ... 3446
Bagaimana cara memantau tindakan yang diambil oleh aturan siklus hidup saya? ... 3446
Jumlah objek S3 saya masih meningkat, bahkan setelah menyiapkan aturan siklus hidup pada bucket yang mendukung versi. ... 3447
Bagaimana cara mengosongkan bucket S3 saya dengan menggunakan aturan siklus hidup? ... 3448

Tagihan Amazon S3 saya meningkat setelah mentransisikan objek ke kelas penyimpanan berbiaya lebih rendah. ... 3449
Saya telah memperbarui kebijakan bucket saya, tetapi objek S3 saya masih dihapus oleh aturan siklus hidup yang kedaluwarsa. ... 3450
Dapatkah saya memulihkan objek S3 yang dikahiri oleh aturan Siklus Hidup S3? ... 3450
Pecahkan masalah replikasi ... 3451
Kiat pemecahan masalah untuk Replikasi S3 ... 3451
Kesalahan Replikasi Batch ... 3457
Memecahkan masalah pencatatan akses server ... 3458
Pesan kesalahan umum saat mengatur pencatatan ... 3458
Memecahkan masalah kegagalan pengiriman ... 3459
Pecahkan masalah Penentuan Versi ... 3461
Saya ingin memulihkan objek yang secara tidak sengaja dihapus dalam bucket dengan Penentuan Versi diaktifkan ... 3461
Saya ingin menghapus objek berversi secara permanen ... 3463
Saya mengalami penurunan kinerja setelah mengaktifkan Penentuan Versi bucket ... 3464
Dapatkan ID permintaan Amazon S3 untuk AWS Support ... 3466
Menggunakan HTTP untuk memperoleh ID Permintaan ... 3466
Menggunakan peramban web untuk memperoleh ID permintaan ... 3467
Menggunakan AWS SDK untuk mendapatkan ID permintaan ... 3467
Menggunakan AWS CLI untuk mendapatkan ID permintaan ... 3470
Menggunakan Windows PowerShell untuk mendapatkan ID permintaan ... 3470
Menggunakan peristiwa AWS CloudTrail data untuk mendapatkan ID permintaan ... 3470
Menggunakan pencatatan log akses server S3 untuk mendapatkan ID permintaan ... 3470
Riwayat dokumen ... 3471
Pembaruan sebelumnya ... 3505
AWSGlosarium ... 3533
... mmmdxxxiv

Apa itu Amazon S3?

Amazon Simple Storage Service (Amazon S3) adalah layanan penyimpanan objek yang menawarkan skalabilitas, ketersediaan data, keamanan, dan kinerja terdepan di industri. Pelanggan dari semua ukuran dan industri dapat menggunakan Amazon S3 untuk menyimpan dan melindungi sejumlah data untuk berbagai kasus penggunaan, seperti danau data, situs web, aplikasi seluler, pencadangan dan pemulihan, arsip, aplikasi perusahaan, perangkat IoT, dan analitik big data. Amazon S3 menyediakan fitur manajemen sehingga Anda dapat mengoptimalkan, mengatur, dan mengonfigurasi akses ke data Anda untuk memenuhi persyaratan bisnis, organisasi, dan kepatuhan spesifik Anda.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Topik

- Fitur-fitur Amazon S3
- Cara kerja Amazon S3
- Model konsistensi data Amazon S3
- Layanan terkait
- Mengakses Amazon S3
- Membayar Amazon S3
- Kepatuhan PCI DSS

Fitur-fitur Amazon S3

Kelas penyimpanan

Amazon S3 menawarkan serangkaian kelas penyimpanan yang dirancang untuk berbagai kasus penggunaan. Misalnya, Anda dapat menyimpan data produksi mission-critical di S3 Standard atau S3 Express One Zone untuk akses yang sering, menghemat biaya dengan menyimpan data yang jarang diakses di S3 Standard-IA atau S3 One Zone-IA, dan mengarsipkan data dengan biaya terendah di S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval, dan S3 Glacier Deep Archive.

Amazon S3 Express One Zone adalah kelas penyimpanan Amazon S3 tunggal berkinerja tinggi yang dibuat khusus untuk menghadirkan akses data milidetik satu digit yang konsisten untuk aplikasi Anda yang paling sensitif terhadap latensi. S3 Express One Zone adalah kelas penyimpanan objek cloud latensi terendah yang tersedia saat ini, dengan kecepatan akses data hingga 10x lebih cepat dan dengan biaya permintaan 50 persen lebih rendah dari Standar S3. S3 Express One Zone adalah kelas penyimpanan S3 pertama di mana Anda dapat memilih satu Zona Ketersediaan dengan opsi untuk menempatkan bersama penyimpanan objek dengan sumber daya komputasi Anda, yang memberikan kecepatan akses setinggi mungkin. Selain itu, untuk lebih meningkatkan kecepatan akses dan mendukung ratusan ribu permintaan per detik, data disimpan dalam jenis bucket baru: bucket direktori Amazon S3. Untuk informasi selengkapnya, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Anda dapat menyimpan data dengan pola akses yang berubah atau tidak dikenal di S3 Intelligent-Tiering, yang mengoptimalkan biaya penyimpanan dengan memindahkan data Anda secara otomatis di antara empat tingkatan akses saat pola akses Anda berubah. Keempat jenjang akses ini mencakup dua jenjang akses latensi rendah yang dioptimalkan untuk akses rutin dan tidak rutin, serta dua jenjang akses arsip pilihan yang dirancang untuk akses asinkron untuk data yang jarang diakses.

Untuk informasi selengkapnya, lihat Menggunakan kelas penyimpanan Amazon S3.

Manajemen penyimpanan

Amazon S3 memiliki fitur manajemen penyimpanan yang dapat Anda gunakan untuk mengelola biaya, memenuhi persyaratan peraturan, mengurangi latensi, dan menyimpan beberapa salinan data Anda yang berbeda untuk persyaratan kepatuhan.

- Siklus Hidup S3–Konfigurasikan konfigurasi siklus hidup untuk mengelola objek Anda dan menyimpannya secara hemat di sepanjang siklus hidupnya. Anda dapat mentransisikan objek ke kelas penyimpanan S3 lainnya atau objek kedaluwarsa yang mencapai akhir masa pakainya.
- Kunci Objek S3–Cegah objek Amazon S3 dihapus atau ditimpa selama jangka waktu tertentu atau tanpa batas waktu. Anda dapat menggunakan Object Lock untuk membantu memenuhi persyaratan peraturan yang memerlukan penyimpanan write-once-read-many(WORM) atau hanya menambahkan lapisan perlindungan lain terhadap perubahan dan penghapusan objek.
- Replikasi S3 - Replikasi objek dan metadata masing-masing dan tag objek ke satu atau beberapa bucket tujuan yang sama atau berbeda Wilayah AWS untuk mengurangi latensi, kepatuhan, keamanan, dan kasus penggunaan lainnya.

- Operasi Batch S3–Kelola miliaran objek dalam skala besar dengan satu permintaan API S3 atau beberapa klik di konsol Amazon S3. Anda dapat menggunakan Operasi Batch untuk melakukan operasi seperti Salin, Panggil fungsi AWS Lambda, dan Pulihkan pada jutaan atau miliaran objek.

Manajemen akses dan keamanan

Amazon S3 menyediakan fitur untuk mengaudit dan mengelola akses ke bucket dan objek Anda. Secara default, bucket S3 dan objek di dalamnya bersifat pribadi. Anda hanya memiliki akses ke sumber daya S3 yang Anda buat. Untuk memberikan izin sumber daya terperinci yang mendukung kasus penggunaan spesifik Anda atau untuk mengaudit izin sumber daya Amazon S3, Anda dapat menggunakan fitur berikut.

- Blokir Akses Publik S3 – Memblokir akses publik ke bucket dan objek S3. Secara default, pengaturan Blokir Akses Publik diaktifkan di tingkat bucket. Kami menyarankan Anda untuk tetap mengaktifkan semua pengaturan Blokir Akses Publik kecuali Anda tahu bahwa Anda perlu mematikan satu atau beberapa pengaturan untuk kasus penggunaan spesifik Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi pengaturan blokir akses publik untuk bucket S3 Anda.
- AWS Identity and Access Management (IAM) - IAM adalah layanan web yang membantu Anda mengontrol akses ke AWS sumber daya dengan aman, termasuk sumber daya Amazon S3 Anda. Dengan IAM, Anda dapat mengelola izin secara terpusat yang mengontrol AWS sumber daya mana yang dapat diakses pengguna. Anda menggunakan IAM untuk mengontrol siapa yang diautentikasi (masuk) dan diotorisasi (memiliki izin) untuk menggunakan sumber daya.
- Kebijakan bucket–Gunakan bahasa kebijakan berbasis IAM untuk mengonfigurasi izin berbasis sumber daya untuk bucket S3 dan objek di dalamnya.
- Titik akses Amazon S3–Konfigurasi titik akhir jaringan bernama dengan kebijakan akses khusus untuk mengelola akses data dalam skala besar untuk set data bersama di Amazon S3.
- Daftar kontrol akses (ACL)–Berikan izin baca dan tulis untuk masing-masing bucket dan objek kepada pengguna yang berwenang. Sebagai aturan umum, sebaiknya gunakan kebijakan berbasis sumber daya S3 (kebijakan bucket dan kebijakan titik akses) atau kebijakan pengguna IAM untuk kontrol akses, bukan ACL. Kebijakan adalah opsi kontrol akses yang disederhanakan dan lebih fleksibel. Dengan kebijakan bucket dan kebijakan titik akses, Anda dapat menentukan aturan yang berlaku secara luas di semua permintaan ke sumber daya Amazon S3 Anda. Untuk informasi selengkapnya tentang kasus spesifik saat Anda menggunakan ACL, bukan kebijakan berbasis sumber daya atau kebijakan pengguna IAM, lihat. Mengelola Akses dengan ACL

- Kepemilikan Objek S3–Ambil kepemilikan setiap objek di bucket Anda, sederhanakan manajemen akses untuk data yang disimpan di Amazon S3. Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk menonaktifkan atau mengaktifkan ACL. Secara default, ACL dinonaktifkan. Ketika ACL dinonaktifkan, pemilik bucket memiliki semua objek di bucket dan mengelola akses ke data secara eksklusif dengan menggunakan kebijakan manajemen akses.
- IAM Access Analyzer for S3–Mengevaluasi dan memantau kebijakan akses bucket S3 Anda, memastikan bahwa kebijakan hanya menyediakan akses yang dimaksudkan ke sumber daya S3 Anda.

Pemrosesan data

Untuk mengubah data dan memicu alur kerja untuk mengotomatiskan berbagai aktivitas pemrosesan lainnya dalam skala besar, Anda dapat menggunakan fitur berikut.

- Lambda Objek S3–Tambahkan kode Anda sendiri ke permintaan S3 GET, HEAD, dan LIST untuk memodifikasi dan memproses data saat dikembalikan ke aplikasi. Filter baris, ubah ukuran gambar secara dinamis, edit data rahasia, dan banyak lagi.
- Pemberitahuan peristiwa - Memicu alur kerja yang menggunakan Amazon Simple Notification Service (Amazon SNS), Amazon Simple Queue Service (Amazon SQS), AWS Lambda dan saat perubahan dilakukan pada sumber daya S3 Anda.

Pencatatan dan pemantauan penyimpanan

Amazon S3 menyediakan alat pencatatan dan pemantauan yang dapat Anda gunakan untuk memantau dan mengontrol penggunaan sumber daya Amazon S3. Untuk informasi selengkapnya, lihat Alat pemantauan.

Alat pemantauan otomatis

- CloudWatchMetrik Amazon untuk Amazon S3 — Lacak kesehatan operasional sumber daya S3 Anda dan konfigurasikan peringatan penagihan saat perkiraan biaya mencapai ambang batas yang ditentukan pengguna.
- AWS CloudTrail— Rekam tindakan yang diambil oleh pengguna, peran, atau Layanan AWS di Amazon S3. CloudTrail log memberi Anda pelacakan API terperinci untuk operasi tingkat ember dan tingkat objek S3.

Alat pemantauan manual

- Pencatatan akses server–menyediakan catatan terperinci untuk permintaan yang dilakukan ke bucket Anda. Anda dapat menggunakan log akses server untuk banyak kasus penggunaan, seperti melakukan keamanan dan audit akses, mempelajari basis pelanggan Anda, dan memahami tagihan Amazon S3.
- AWS Trusted Advisor - Evaluasi akun Anda dengan menggunakan pemeriksaan praktik AWS terbaik untuk mengidentifikasi cara mengoptimalkan AWS infrastruktur Anda, meningkatkan keamanan dan kinerja, mengurangi biaya, dan memantau kuota layanan. Anda kemudian dapat mengikuti rekomendasi untuk mengoptimalkan layanan dan sumber daya Anda.

Analitik dan wawasan

Amazon S3 menawarkan fitur untuk membantu Anda mendapatkan visibilitas ke dalam penggunaan penyimpanan, yang memberdayakan Anda untuk lebih memahami, menganalisis, dan mengoptimalkan penyimpanan Anda dalam skala besar.

- Lensa Penyimpanan Amazon S3–Memahami, menganalisis, dan mengoptimalkan penyimpanan Anda. S3 Storage Lens menyediakan 60+ metrik penggunaan dan aktivitas serta dasbor interaktif untuk mengumpulkan data untuk seluruh organisasi, akun tertentu, bucket, atau awalan Anda. Wilayah AWS
- Analisis Kelas Penyimpanan–Analisis pola akses penyimpanan untuk memutuskan kapan saatnya memindahkan data ke kelas penyimpanan yang lebih hemat biaya.
- Inventaris S3 dengan laporan Inventaris–Audit dan laporkan objek dan metadata terkait serta konfigurasikan fitur Amazon S3 lainnya untuk mengambil tindakan dalam laporan Inventaris. Misalnya, Anda dapat melaporkan replikasi dan status enkripsi objek Anda. Untuk daftar semua metadata yang tersedia untuk setiap objek dalam laporan Inventaris, lihat daftar Inventaris Amazon S3.

Konsistensi kuat

Amazon S3 memberikan read-after-write konsistensi yang kuat untuk permintaan PUT dan DELETE objek di bucket Amazon S3 Anda secara keseluruhan. Wilayah AWS Perilaku ini berlaku untuk penulisan objek baru dan permintaan PUT yang menimpa objek yang sudah ada dan permintaan DELETE. Selain itu, operasi baca di Amazon S3 Select, daftar kontrol akses Amazon S3 (ACL),

Amazon S3 Object Tag, dan metadata objek (misalnya, objek HEAD) sangat konsisten. Untuk informasi selengkapnya, lihat Model konsistensi data Amazon S3.

Cara kerja Amazon S3

Amazon S3 adalah layanan penyimpanan objek yang menyimpan data sebagai objek dalam bucket. Objek adalah file data dan metadata apa pun yang mendeskripsikan file tersebut. Bucket adalah kontainer untuk objek.

Untuk menyimpan data Anda di Amazon S3, Anda terlebih dahulu membuat bucket dan menentukan nama bucket dan Wilayah AWS. Kemudian, Anda mengunggah data Anda ke bucket itu sebagai objek di Amazon S3. Setiap objek memiliki kunci (atau nama kunci), yang merupakan pengidentifikasi unik untuk objek di dalam bucket.

S3 menyediakan fitur yang dapat Anda konfigurasi untuk mendukung kasus penggunaan spesifik Anda. Sebaga contoh, Anda dapat menggunakan Penentuan Versi S3 untuk menyimpan beberapa versi dari sebuah objek di dalam satu bucket sehingga Anda dapat memulihkan objek yang tidak sengaja terhapus atau tertimpa.

Bucket dan objek di dalamnya bersifat pribadi dan hanya dapat diakses jika Anda secara eksplisit memberikan izin akses. Anda dapat menggunakan kebijakan bucket, kebijakan AWS Identity and Access Management (IAM), daftar kontrol akses (ACL), dan Titik Akses S3 untuk mengelola akses.

Topik

- Bucket
- Objek
- Kunci
- Penentuan Versi S3
- ID Versi
- Kebijakan bucket
- Titik Akses S3
- Daftar kontrol akses (ACL)
- Wilayah

Bucket

Bucket adalah kontainer untuk objek yang disimpan dalam Amazon S3. Anda dapat menyimpan berapa pun jumlah objek dalam bucket dan dapat memiliki hingga 100 bucket di akun Anda. Untuk meminta peningkatan, kunjungi Konsol Kuota Layanan.

Setiap objek dimuat dalam bucket. Misalnya, jika objek bernama `photos/puppy.jpg` disimpan dalam bucket `DOC-EXAMPLE-BUCKET` di Wilayah AS Barat (Oregon), maka objek tersebut dapat dialamatkan dengan menggunakan URL `https://DOC-EXAMPLE-BUCKET.s3.us-west-2.amazonaws.com/photos/puppy.jpg`. Untuk informasi lebih lanjut, lihat Mengakses Bucket.

Saat Anda membuat bucket, memasukkan nama bucket dan memilih Wilayah AWS tempat bucket tersebut akan berada. Setelah membuat bucket, Anda tidak dapat mengubah nama atau Wilayah-nya. Nama bucket harus mengikuti aturan penamaan bucket. Anda juga dapat mengonfigurasi bucket agar menggunakan Penentuan Versi S3 atau fitur manajemen penyimpanan lainnya.

Bucket juga:

- Mengatur ruang nama Amazon S3 di tingkat tertinggi.
- Mengidentifikasi akun yang bertanggung jawab atas biaya penyimpanan dan transfer data.
- Berikan opsi kontrol akses, seperti kebijakan bucket, daftar kontrol akses (ACL), dan Titik Akses S3, yang dapat Anda gunakan untuk mengelola akses ke sumber daya Amazon S3.
- Berfungsi sebagai unit agregasi untuk pelaporan penggunaan.

Untuk informasi selengkapnya tentang bucket, lihat Gambaran umum bucket.

Objek

Objek adalah entitas dasar yang disimpan di Amazon S3. Objek terdiri atas data objek dan metadata. Metadata adalah serangkaian pasangan nilai-nama yang menjelaskan objek. Pasangan ini mencakup beberapa metadata default, seperti tanggal terakhir diubah, dan metadata HTTP standar, seperti `Content-Type`. Anda juga dapat menentukan metadata kustom pada saat objek disimpan.

Objek diidentifikasi secara unik dalam bucket dengan kunci (nama) dan ID versi (jika Penentuan Versi S3 diaktifkan di bucket). Untuk informasi selengkapnya tentang objek, lihat Gambaran umum objek Amazon S3.

Kunci

Kunci objek (atau nama kunci) adalah pengidentifikasi unik untuk objek dalam bucket. Setiap objek dalam bucket memiliki persis satu kunci. Kombinasi bucket, kunci objek, dan ID versi opsional (jika Penentuan Versi S3 diaktifkan untuk bucket) secara unik mengidentifikasi setiap objek. Jadi Anda dapat membayangkan Amazon S3 sebagai peta data dasar antara "bucket + kunci + versi" dan objek itu sendiri.

Setiap objek di Amazon S3 dapat dialamatkan secara unik melalui kombinasi titik akhir layanan web, nama bucket, kunci, dan secara opsional, versi. Misalnya, di URL `https://DOC-EXAMPLE-BUCKET.s3.us-west-2.amazonaws.com/photos/puppy.jpg`, `DOC-EXAMPLE-BUCKET` adalah nama bucket dan `photos/puppy.jpg` merupakan kuncinya.

Untuk informasi selengkapnya tentang kunci objek, lihat Membuat nama kunci objek.

Penentuan Versi S3

Anda dapat menggunakan Penentuan Versi S3 untuk menyimpan beberapa varian objek dalam bucket yang sama. Dengan Penentuan Versi S3, Anda dapat menyimpan, mengambil, dan memulihkan setiap versi dari setiap objek yang disimpan dalam bucket Anda. Anda dapat lebih mudah memulihkan dari tindakan pengguna yang tidak diinginkan dan kegagalan aplikasi.

Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.

ID Versi

Saat Anda mengaktifkan Penentuan Versi S3 dalam bucket, S3 Amazon menghasilkan ID versi unik untuk setiap objek yang ditambahkan ke bucket. Objek yang sudah ada di bucket pada saat Anda mengaktifkan Penentuan Versi memiliki ID versi . Jika Anda memodifikasi objek ini (atau lainnya) dengan operasi lain, seperti CopyObjectdan PutObject, objek baru mendapatkan ID versi unik.

Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.

Kebijakan bucket

Kebijakan bucket adalah kebijakan berbasis sumber daya AWS Identity and Access Management (IAM) yang dapat Anda gunakan untuk memberikan izin akses ke bucket dan objek di dalamnya. Hanya pemilik bucket yang dapat mengaitkan kebijakan dengan bucket. Izin yang dipasang pada

bucket berlaku untuk semua objek di bucket yang dimiliki oleh pemilik bucket. Kebijakan bucket dibatasi hingga ukuran 20 KB.

Kebijakan bucket menggunakan bahasa kebijakan akses berbasis JSON yang standar di seluruh AWS. Anda dapat menggunakan kebijakan bucket untuk menambah atau menolak izin objek dalam bucket. Kebijakan bucket mengizinkan atau menolak permintaan berdasarkan elemen-elemen dalam kebijakan, termasuk pemohon, tindakan S3, sumber daya, dan aspek atau kondisi permintaan (misalnya, alamat IP yang digunakan untuk membuat permintaan). Misalnya, Anda dapat membuat kebijakan bucket yang memberikan izin lintas akun untuk mengunggah objek ke bucket S3 sekaligus memastikan pemilik bucket memiliki kendali penuh atas objek yang diunggah. Untuk informasi selengkapnya, lihat Contoh kebijakan bucket Amazon S3.

Dalam kebijakan bucket, Anda dapat menggunakan karakter wildcard pada Amazon Resource Name (ARN) dan nilai lainnya untuk memberikan izin ke subset objek. Misalnya, Anda dapat mengendalikan akses ke kelompok objek yang dimulai dengan prefiks umum atau diakhiri dengan ekstensi tertentu, seperti `.html`.

Titik Akses S3

Titik Akses Amazon S3 diberi nama titik akhir jaringan dengan kebijakan akses khusus yang menjelaskan cara data dapat diakses menggunakan titik akhir tersebut. Access Points dilampirkan ke bucket yang dapat Anda gunakan untuk melakukan operasi objek S3, seperti GetObject dan. PutObject Titik akses menyederhanakan pengelolaan akses data dalam skala besar untuk set data bersama di Amazon S3.

Setiap titik akses memiliki kebijakan jalur aksesnya sendiri. Anda dapat mengonfigurasi pengaturan Blokir Akses Publik untuk setiap titik akses. Untuk membatasi akses data Amazon S3 ke jaringan privat, Anda juga dapat mengonfigurasi titik akses apa pun untuk menerima permintaan hanya dari cloud privat virtual (VPC).

Untuk informasi selengkapnya, lihat Mengelola akses data dengan titik akses Amazon S3.

Daftar kontrol akses (ACL)

Anda dapat menggunakan ACL untuk memberikan izin baca dan tulis kepada pengguna yang berwenang untuk masing-masing bucket dan objek. Setiap bucket dan objek memiliki ACL yang melekat padanya sebagai sumber daya tambahan. ACL mendefinisikan Akun AWS atau kelompok mana yang diberikan akses dan jenis akses. ACL adalah mekanisme kontrol akses yang berlaku di awal IAM. Untuk informasi selengkapnya tentang ACL, lihat Gambaran umum daftar kontrol akses (ACL).

Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda, serta menonaktifkan atau mengaktifkan ACL. Secara default, Kepemilikan Objek diatur ke pengaturan yang diberlakukan pemilik Bucket, dan semua ACL dinonaktifkan. Ketika ACL dinonaktifkan, pemilik bucket memiliki semua objek di bucket dan mengelola akses ke objek-objek tersebut secara eksklusif dengan menggunakan kebijakan manajemen akses.

Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL. Kami menyarankan agar ACL dinonaktifkan, kecuali dalam keadaan yang tidak biasa ketika Anda perlu mengontrol akses untuk setiap objek secara individual. Dengan ACL dinonaktifkan, Anda dapat menggunakan kebijakan untuk mengontrol akses ke semua objek di bucket, terlepas dari siapa yang mengunggah objek ke bucket Anda. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Wilayah

Anda dapat memilih geografis Wilayah AWS tempat Amazon S3 menyimpan ember yang Anda buat. Anda dapat memilih Wilayah untuk mengoptimalkan latensi, meminimalkan biaya, atau memenuhi persyaratan peraturan. Objek yang disimpan di Wilayah AWS tidak pernah meninggalkan Wilayah kecuali Anda secara eksplisit mentransfer atau mereplikasi mereka ke Wilayah lain. Misalnya, objek yang disimpan di Wilayah Eropa (Irlandia) tidak pernah keluar dari Wilayah tersebut.

> **Note**
>
> Anda dapat mengakses Amazon S3 dan fitur-fiturnya hanya di Wilayah AWS yang diaktifkan untuk akun Anda. Untuk informasi selengkapnya tentang mengaktifkan Wilayah untuk membuat dan mengelola AWS sumber daya, lihat Mengelola Wilayah AWS di. Referensi Umum AWS

Untuk daftar titik akhir dan Wilayah Amazon S3, lihat Titik akhir dan Wilayah di Referensi Umum AWS.

Model konsistensi data Amazon S3

Amazon S3 memberikan read-after-write konsistensi yang kuat untuk permintaan PUT dan DELETE objek di bucket Amazon S3 Anda secara keseluruhan. Wilayah AWS Perilaku ini berlaku untuk

penulisan objek baru dan permintaan PUT yang menimpa objek yang sudah ada dan permintaan DELETE. Selain itu, operasi baca di Amazon S3 Select, daftar kontrol akses (ACL) Amazon S3, Amazon S3 Object Tag, dan metadata objek (misalnya, objek HEAD) sangat konsisten.

Pembaruan untuk satu kunci bersifat atomik. Misalnya, jika Anda membuat permintaan PUT ke kunci yang sudah ada dari satu utas dan melakukan permintaan GET pada kunci yang sama dari utas kedua secara bersamaan, Anda akan mendapatkan data lama atau data baru, tetapi tidak pernah sebagian atau data yang rusak.

Amazon S3 mencapai ketersediaan tinggi dengan mereplikasi data di beberapa server di pusat data AWS . Jika permintaan PUT berhasil, data Anda disimpan dengan aman. Setiap pembacaan (permintaan GET atau LIST) yang dimulai setelah penerimaan respons PUT yang berhasil akan mengembalikan data yang ditulis oleh permintaan PUT. Berikut adalah contoh dari perilaku ini:

- Proses menulis objek baru ke Amazon S3 dan segera mencantumkan kunci di dalam bucket-nya. Objek baru akan muncul dalam daftar.
- Proses menggantikan objek yang sudah ada dan segera mencoba membacanya. Amazon S3 mengembalikan data baru.
- Proses menggantikan objek yang sudah ada dan segera mencoba membacanya. Amazon S3 tidak mengembalikan data apa pun karena objek telah dihapus.
- Proses menghapus objek yang sudah ada dan segera mencantumkan kunci di dalam bucket-nya. Objek tidak muncul dalam daftar.

> **Note**
>
> - Amazon S3 tidak mendukung penguncian objek untuk penulisan bersamaan. Jika dua permintaan PUT secara bersamaan dibuat ke kunci yang sama, permintaan dengan stempel waktu terbaru akan menang. Jika ini menjadi masalah, Anda harus membangun mekanisme penguncian objek ke dalam aplikasi Anda.
> - Pembaruan bersifat berbasis kunci. Tidak ada cara untuk melakukan pembaruan atomik lintas kunci. Sebagai contoh, Anda tidak dapat memperbarui satu kunci menjadi tergantung pada pembaruan kunci lain kecuali Anda merancang fungsionalitas ini ke dalam aplikasi Anda.

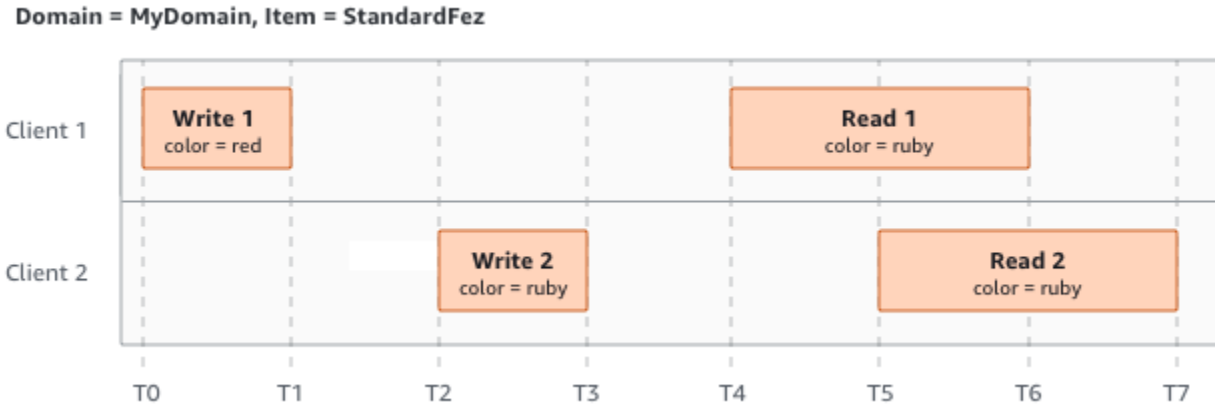
Konfigurasi bucket memiliki model konsistensi pada akhir. Secara khusus, ini berarti:

- Jika Anda menghapus bucket dan langsung membuat daftar semua bucket, bucket yang dihapus mungkin masih akan muncul dalam daftar.
- Jika Anda mengaktifkan Penentuan Versi pada bucket untuk pertama kalinya, mungkin diperlukan beberapa saat agar perubahan dapat dilakukan sepenuhnya. Kami menyarankan Anda menunggu selama 15 menit setelah mengaktifkan Penentuan Versi sebelum memberikan operasi tulis (PUT atau DELETE) pada objek di dalam bucket.

Aplikasi yang bersamaan

Bagian ini memberikan contoh perilaku yang diharapkan dari Amazon S3 ketika beberapa klien menulis ke item yang sama.

Dalam contoh ini, W1 (tulis 1) dan W2 (tulis 2) selesai sebelum awal R1 (baca 1) dan R2 (baca 2). Karena S3 sangat konsisten, R1 dan R2 keduanya mengembalikan `color = ruby`.



Dalam contoh berikutnya, W2 tidak selesai sebelum awal R1. Oleh karena itu, R1 mungkin mengembalikan `color = ruby` atau `color = garnet`. Namun, karena W1 dan W2 selesai sebelum dimulainya R2, R2 mengembalikan `color = garnet`.

Dalam contoh terakhir, W2 dimulai sebelum W1 telah menerima pengakuan. Oleh karena itu, penulisan ini dianggap bersamaan. Amazon S3 secara internal menggunakan last-writer-wins semantik untuk menentukan penulisan mana yang diutamakan. Namun, urutan di mana Amazon S3 menerima permintaan dan urutan aplikasi menerima pengakuan tidak dapat diprediksi karena berbagai faktor, seperti latensi jaringan. Misalnya, W2 mungkin dimulai oleh instans Amazon EC2 di Wilayah yang sama, sementara W1 mungkin dimulai oleh host yang lebih jauh. Cara terbaik untuk menentukan nilai akhir adalah dengan melakukan pembacaan setelah kedua penulisan telah diakui.

Layanan terkait

Setelah memuat data ke Amazon S3, Anda dapat menggunakannya dengan layanan lain AWS . Berikut adalah layanan yang mungkin paling sering Anda gunakan:

- Amazon Elastic Compute Cloud (Amazon EC2)–Menyediakan kapasitas komputasi yang aman dan dapat diskalakan di AWS Cloud. Dengan menggunakan Amazon EC2, Anda tidak perlu berinvestasi pada perangkat keras di awal, sehingga Anda bisa mengembangkan dan men-deploy

aplikasi lebih cepat. Anda dapat menggunakan Amazon EC2 untuk meluncurkan server virtual sebanyak atau sesedikit yang Anda butuhkan, mengonfigurasi keamanan dan jaringan, serta mengelola penyimpanan.

- Amazon EMR–Membantu bisnis, peneliti, analis data, dan pengembang memproses data dalam jumlah besar dengan mudah dan hemat biaya. Amazon EMR menggunakan kerangka kerja Hadoop yang dihosting yang berjalan di infrastruktur skala web Amazon EC2 dan Amazon S3.
- AWS Keluarga Salju — Membantu pelanggan yang perlu menjalankan operasi di lingkungan non-pusat data yang keras, dan di lokasi di mana ada kurangnya konektivitas jaringan yang konsisten. Anda dapat menggunakan perangkat AWS Snow Family untuk mengakses penyimpanan dan daya komputasi penyimpanan secara lokal dan hemat biaya AWS Cloud di tempat-tempat di mana koneksi internet mungkin bukan pilihan.
- AWS Transfer Family–Menyediakan dukungan yang dikelola sepenuhnya untuk transfer file langsung masuk dan keluar dari Amazon S3 atau Amazon Elastic File System (Amazon EFS) menggunakan Secure Shell (SSH) File Transfer Protocol (SFTP), File Transfer Protocol melalui SSL (FTPS), dan File Transfer Protocol (FTP).

Mengakses Amazon S3

Anda dapat bekerja dengan Amazon S3 dengan salah satu cara berikut ini:

AWS Management Console

Konsol adalah antarmuka pengguna berbasis web untuk mengelola Amazon S3 AWS dan sumber daya. Jika Anda telah mendaftar Akun AWS, Anda dapat mengakses konsol Amazon S3 dengan masuk ke AWS Management Console dan memilih S3 dari halaman beranda. AWS Management Console

AWS Command Line Interface

Anda dapat menggunakan alat baris AWS perintah untuk mengeluarkan perintah atau membangun skrip di baris perintah sistem Anda untuk melakukan tugas AWS (termasuk S3).

The AWS Command Line Interface (AWS CLI) menyediakan perintah untuk satu set yang luas Layanan AWS. AWS CLI Ini didukung di Windows, macOS, dan Linux. Untuk memulai, lihat Panduan Pengguna AWS Command Line Interface . Untuk informasi selengkapnya tentang perintah untuk Amazon S3, lihat s3api dan s3control di AWS CLI Referensi Perintah.

AWS SDK

AWS menyediakan SDK (perangkat pengembangan perangkat lunak) yang terdiri dari pustaka dan kode sampel untuk berbagai bahasa dan platform pemrograman (Java, Python, Ruby, .NET, iOS, Android, dan sebagainya). AWS SDK menyediakan cara mudah untuk membuat akses terprogram ke S3 dan. AWS Amazon S3 adalah layanan REST. Anda dapat mengirim permintaan ke Amazon S3 menggunakan pustaka AWS SDK, yang membungkus API REST Amazon S3 yang mendasarinya dan menyederhanakan tugas pemrograman Anda. Misalnya, SDK menangani tugas seperti menghitung tanda tangan, menandatangani permintaan secara kriptografis, mengelola kesalahan, dan mencoba kembali permintaan secara otomatis. Untuk informasi tentang AWS SDK, termasuk cara mengunduh dan menginstalnya, lihat Alat untuk AWS.

Setiap interaksi dengan Amazon S3 diautentikasi atau dilakukan secara anonim. Jika Anda menggunakan AWS SDK, pustaka menghitung tanda tangan untuk otentikasi dari kunci yang Anda berikan. Untuk informasi selengkapnya tentang cara membuat permintaan ke Amazon S3, lihat. Membuat permintaan

API REST Amazon S3

Arsitektur Amazon S3 dirancang untuk menjadi bahasa pemrograman-netral, menggunakan antarmuka yang didukung AWS untuk menyimpan dan mengambil objek. Anda dapat mengakses S3 dan AWS secara terprogram menggunakan API REST Amazon S3. API REST adalah antarmuka HTTP ke Amazon S3. Dengan API REST, Anda menggunakan permintaan HTTP standar untuk membuat, dan menghapus bucket dan objek.

Untuk menggunakan API REST, Anda dapat menggunakan toolkit yang mendukung HTTP. Anda bahkan dapat menggunakan peramban untuk mengambil objek, selama objek tersebut dapat dibaca secara anonim.

API REST menggunakan header HTTP standar dan kode status, sehingga peramban dan toolkit standar bekerja sesuai harapan. Di beberapa area, kami telah menambahkan fungsi ke HTTP (misalnya, kami menambahkan header untuk mendukung kontrol akses). Dalam kasus ini, kami telah melakukan yang terbaik untuk menambahkan fungsionalitas baru dengan cara yang cocok dengan cara penggunaan HTTP standar.

Jika Anda melakukan panggilan API REST langsung di aplikasi Anda, Anda harus menulis kode untuk menghitung tanda tangan dan menambahkannya ke permintaan. Untuk informasi selengkapnya tentang cara membuat permintaan ke Amazon S3, lihat. Membuat permintaan

> **Note**
>
> Dukungan SOAP API melalui HTTP dihilangkan, tetapi masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak didukung untuk SOAP. Kami menyarankan Anda menggunakan REST API atau AWS SDK.

Membayar Amazon S3

Harga untuk Amazon S3 dirancang agar Anda tidak perlu merencanakan persyaratan penyimpanan aplikasi Anda. Sebagian besar penyedia penyimpanan mengharuskan Anda untuk membeli kapasitas penyimpanan dan transfer jaringan dalam jumlah yang telah ditentukan sebelumnya. Dalam skenario ini, jika Anda melebihi kapasitas tersebut, layanan Anda akan dimatikan atau Anda akan dikenai biaya kelebihan penggunaan yang tinggi. Jika Anda tidak melebihi kapasitas tersebut, Anda membayar seolah-olah Anda menggunakan semuanya.

Amazon S3 mengenakan biaya hanya untuk yang benar-benar Anda gunakan, tanpa biaya tersembunyi dan biaya kelebihan. Model ini memberi Anda layanan biaya variabel yang dapat tumbuh dengan bisnis Anda sambil memberi Anda keuntungan biaya infrastruktur. AWS Untuk informasi selengkapnya, lihat harga Amazon S3.

Ketika Anda mendaftar AWS, Anda Akun AWS secara otomatis mendaftar untuk semua layanan di AWS, termasuk Amazon S3. Nmaun, Anda hanya dikenai biaya untuk layanan yang digunakan. Jika Anda adalah pelanggan baru Amazon S3, Anda dapat memulai Amazon S3 secara gratis. Untuk informasi selengkapnya, lihat AWS tingkat gratis.

Untuk melihat tagihan Anda, buka Dasbor Manajemen Penagihan dan Biaya di konsol AWS Billing and Cost Management. Untuk mempelajari selengkapnya tentang Akun AWS penagihan, lihat Panduan AWS Billing Pengguna. Jika Anda memiliki pertanyaan tentang AWS penagihan dan Akun AWS, hubungi AWS Support.

Kepatuhan PCI DSS

Amazon S3 mendukung pemrosesan, penyimpanan, dan transmisi data kartu kredit oleh pedagang atau penyedia layanan, serta telah divalidasi sesuai dengan Standar Keamanan Data (DSS) Industri Kartu Pembayaran (PCI). Untuk informasi selengkapnya tentang PCI DSS, termasuk cara meminta salinan PCI AWS Compliance Package, lihat PCI DSS Level 1.

Memulai dengan Amazon S3

Anda dapat memulai dengan Amazon S3 dengan bekerja dengan bucket dan objek. Bucket adalah kontainer untuk objek. Objek adalah file data dan metadata apa pun yang mendeskripsikan file tersebut.

Untuk menyimpan objek di Amazon S3, Anda membuat bucket dan kemudian mengunggah objek ke bucket. Bila objek ada di dalam bucket, Anda bisa membuka, mengunduh, dan memindahkannya. Saat Anda tidak lagi memerlukan objek atau bucket, Anda dapat membersihkan sumber daya Anda.

Dengan Amazon S3, Anda hanya membayar sesuai penggunaan Anda. Untuk informasi selengkapnya tentang fitur dan harga Amazon S3, lihat Amazon S3. Jika Anda adalah pelanggan baru Amazon S3, Anda dapat memulai Amazon S3 secara gratis. Untuk informasi selengkapnya, lihat AWS Tingkat Gratis.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Video: Memulai Amazon S3

Prasyarat

Sebelum memulai, konfirmasikan Anda telah menyelesaikan langkah-langkah dalam Prasyarat: Menyiapkan Amazon S3.

Topik

- Prasyarat: Menyiapkan Amazon S3
- Langkah 1: Buat bucket S3 pertama Anda
- Langkah 2: Unggah satu objek ke bucket Anda
- Langkah 3: Mengunduh objek
- Langkah 4: Salin objek ke folder
- Langkah 5: Hapus objek dan bucket Anda

- Langkah selanjutnya

Prasyarat: Menyiapkan Amazon S3

Ketika Anda mendaftar AWS, Anda Akun AWS secara otomatis mendaftar untuk semua layanan di AWS, termasuk Amazon S3. Anda hanya membayar biaya layanan yang Anda gunakan.

Dengan Amazon S3, Anda hanya membayar sesuai penggunaan Anda. Untuk informasi selengkapnya tentang fitur dan harga Amazon S3, lihat Amazon S3. Jika Anda adalah pelanggan baru Amazon S3, Anda dapat memulai Amazon S3 secara gratis. Untuk informasi selengkapnya, lihat AWS Tingkat Gratis.

Untuk mengatur Amazon S3, gunakan langkah-langkah di bagian berikut.

Saat Anda mendaftar AWS dan menyiapkan Amazon S3, Anda dapat mengubah bahasa tampilan secara opsional di. AWS Management Console Untuk informasi lebih lanjut, lihat Mengubah bahasa AWS Management Console di AWS Management Console Panduan memulai.

Topik

- Mendaftar untuk Akun AWS
- Buat pengguna dengan akses administratif

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka https://portal.aws.amazon.com/billing/signup.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan tindakan menerima panggilan telepon dan memasukkan kode verifikasi di keypad telepon.

   Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

AWS mengirimi Anda email konfirmasi setelah proses pendaftaran selesai. Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan mengunjungi https://aws.amazon.com/ dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Anda Pengguna root akun AWS

1. Masuk ke AWS Management Consolesebagai pemilik akun dengan memilih pengguna Root dan memasukkan alamat Akun AWS email Anda. Di laman berikutnya, masukkan kata sandi.

   Untuk bantuan masuk dengan menggunakan pengguna root, lihat Masuk sebagai pengguna root di AWS Sign-In Panduan Pengguna.

2. Mengaktifkan autentikasi multi-faktor (MFA) untuk pengguna root Anda.

   Untuk petunjuk, lihat Mengaktifkan perangkat MFA virtual untuk pengguna Akun AWS root (konsol) Anda di Panduan Pengguna IAM.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat Identitas IAM.

   Untuk mendapatkan petunjuk, silakan lihat Mengaktifkan AWS IAM Identity Center di Panduan Pengguna AWS IAM Identity Center .

2. Di Pusat Identitas IAM, berikan akses administratif ke pengguna.

   Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat Identitas IAM, gunakan URL masuk yang dikirim ke alamat email saat Anda membuat pengguna Pusat Identitas IAM.

Untuk bantuan masuk menggunakan pengguna Pusat Identitas IAM, lihat Masuk ke portal AWS akses di Panduan AWS Sign-In Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat Identitas IAM, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

   Untuk petunjuknya, lihat Membuat set izin di Panduan AWS IAM Identity Center Pengguna.
2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

   Untuk petunjuk, lihat Menambahkan grup di Panduan AWS IAM Identity Center Pengguna.

Langkah 1: Buat bucket S3 pertama Anda

Setelah mendaftar AWS, Anda siap membuat ember di Amazon S3 menggunakan file. AWS Management Console Setiap objek di Amazon S3 disimpan di bucket. Sebelum Anda dapat menyimpan data di Amazon S3, Anda harus membuat bucket.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

> **Note**
>
> Anda tidak dikenakan biaya untuk membuat bucket. Anda hanya dikenakan untuk menyimpan objek di dalam bucket dan untuk memindahkan objek masuk dan keluar dari bucket. Biaya yang Anda tanggung melalui mengikuti contoh dalam panduan ini adalah minimal (kurang dari $1). Untuk informasi selengkapnya tentang biaya penyimpanan, lihat Harga Amazon S3.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di bilah navigasi di bagian atas halaman, pilih nama yang saat ini ditampilkan Wilayah AWS. Selanjutnya, pilih Wilayah tempat Anda ingin membuat ember.

> **Note**
>
> Untuk meminimalkan latensi dan biaya serta memenuhi persyaratan regulasi, pilih Wilayah yang dekat dengan Anda. Objek yang disimpan di Wilayah tidak pernah keluar dari Wilayah kecuali Anda secara tegas mentransfer atau mereplikasinya ke Wilayah lain. Untuk daftar Amazon S3 Wilayah AWS, lihat Layanan AWS titik akhir di. Referensi Umum Amazon Web Services

3. Di panel navigasi kiri, pilih Bucket.
4. Pilih Buat bucket.

   Halaman Buat bucket terbuka.
5. Di bawah Konfigurasi umum, lihat Wilayah AWS tempat bucket Anda akan dibuat.
6. Di bawah jenis Bucket, pilih Tujuan umum.
7. Untuk Nama bucket, masukkan nama untuk bucket Anda.

   Nama bucket harus:

   - Unik dalam partisi. Partisi adalah pembuatan grup Wilayah. Saat ini, AWS memiliki tiga partisi: `aws` (Wilayah Standar), `aws-cn` (Wilayah Tiongkok), dan `aws-us-gov` (AWS GovCloud (US) Regions).
   - Panjangnya antara 3 hingga 63 karakter.
   - Hanya terdiri dari huruf kecil, angka, titik (.), dan tanda hubung (-). Untuk kompatibilitas terbaik, kami menyarankan agar Anda menghindari penggunaan titik (.) dalam nama bucket, kecuali untuk bucket yang digunakan hanya untuk menghosting situs web statis.
   - Dimulai dan diakhiri dengan huruf atau angka.

   Setelah membuat bucket, Anda tidak dapat mengubah namanya. Untuk informasi selengkapnya tentang penamaan bucket, lihat Peraturan penamaan bucket.

> **Important**
>
> Hindari menyertakan informasi sensitif, seperti nomor akun, dalam nama bucket. Nama bucket terlihat dalam URL yang menunjuk objek dalam bucket.

8. AWS Management Console memungkinkan Anda menyalin pengaturan bucket yang ada ke bucket baru Anda. Jika Anda tidak ingin menyalin pengaturan bucket yang ada, lewati ke langkah berikutnya.

   > **Note**
   >
   > Opsi ini:
   >
   > - Tidak tersedia di AWS CLI dan hanya tersedia di konsol
   > - Tidak tersedia untuk bucket direktori
   > - Tidak menyalin kebijakan bucket dari bucket yang ada ke bucket baru

   Untuk menyalin setelan bucket yang ada, di bagian Salin setelan dari bucket yang ada, pilih Pilih bucket. Jendela Select bucket terbuka. Temukan bucket dengan pengaturan yang ingin Anda salin, lalu pilih Pilih bucket. Jendela Choose bucket ditutup, dan jendela Create bucket terbuka kembali.

   Di bawah Salin pengaturan dari bucket yang ada, Anda sekarang akan melihat nama bucket yang Anda pilih. Anda juga akan melihat opsi Restore default yang dapat Anda gunakan untuk menghapus pengaturan bucket yang disalin. Tinjau setelan bucket yang tersisa, di halaman Buat bucket. Anda akan melihat bahwa mereka sekarang cocok dengan pengaturan ember yang Anda pilih. Anda dapat melompat ke langkah terakhir.

9. Di bawah Kepemilikan Objek, untuk menonaktifkan atau mengaktifkan ACL dan mengontrol kepemilikan objek yang diunggah di bucket Anda, pilih salah satu pengaturan berikut ini:

   ACL dinonaktifkan

   - Pemilik bucket yang diberlakukan (default)–ACL dinonaktifkan, dan pemilik bucket secara otomatis memilikinya, serta mendapatkan kontrol penuh atas setiap objek di dalam bucket. ACL tidak lagi memengaruhi izin akses ke data di bucket S3. Bucket menggunakan kebijakan secara eksklusif untuk menentukan kontrol akses.

Secara default, ACL dinonaktifkan. Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL. Kami menyarankan agar ACL dinonaktifkan, kecuali dalam keadaan yang tidak biasa ketika Anda perlu mengontrol akses untuk setiap objek secara individual. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

ACL diaktifkan

- Pemilik bucket yang dipilih—Pemilik bucket memiliki dan diberikan kendali penuh atas objek baru yang ditulis akun lain ke bucket dengan ACL `bucket-owner-full-control` yang dibatasi.

  Jika Anda menerapkan pengaturan Pemilik bucket yang dipilih, agar semua unggahan Amazon S3 menyertakan ACL `bucket-owner-full-control` yang terekam, Anda dapat menambahkan kebijakan bucket yang hanya mengizinkan unggahan objek yang menggunakan ACL ini.
- Penulis objek — Akun AWS Yang mengunggah objek memiliki objek, memiliki kontrol penuh atasnya, dan dapat memberikan pengguna lain akses ke sana melalui ACL.

> **Note**
>
> Pengaturan default-nya adalah Pemilik Bucket yang diberlakukan. Untuk menerapkan pengaturan default dan menonaktifkan ACL, hanya izin `s3:CreateBucket` yang diperlukan. Untuk mengaktifkan ACL, Anda harus memiliki izin `s3:PutBucketOwnershipControls`.

10. Di bawah Pengaturan Blokir Akses Publik untuk bucket ini, pilih pengaturan Blokir Akses Publik yang ingin Anda terapkan ke bucket.

    Secara default, semua pengaturan Blokir Akses Publik untuk bucket direktori diaktifkan. Kami menyarankan Anda tetap mengaktifkan semua pengaturan, kecuali Anda tahu bahwa Anda perlu menonaktifkan satu atau beberapa pengaturan untuk kasus penggunaan spesifik Anda. Untuk informasi lebih lanjut tentang pemblokiran akses publik, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

> **Note**
>
> Untuk mengaktifkan semua pengaturan Blokir Akses Publik, hanya izin `s3:CreateBucket` yang diperlukan. Untuk mematikan pengaturan Blokir Akses Publik, Anda harus memiliki izin `s3:PutBucketPublicAccessBlock`.

11. (Opsional) Di bawah Penentuan Versi Bucket, Anda dapat memilih apakah Anda ingin menyimpan varian objek di bucket Anda. Untuk informasi selengkapnya tentang penentuan versi, lihat Menggunakan Penentuan Versi dalam bucket S3.

    Untuk menonaktifkan atau mengaktifkan Penentuan Versi di bucket Anda, pilih Nonaktifkan atau Aktifkan.

12. (Opsional) Di bawah Tanda, Anda dapat memilih untuk menambahkan tanda ke bucket Anda. Tanda adalah pasangan kunci-nilai yang digunakan untuk mengategorikan penyimpanan.

    Untuk menambahkan tanda bucket, masukkan Kunci dan secara opsional Nilai, lalu pilih Tambahkan Tanda.

13. Di bagian bawah Enkripsi default, pilih Edit.
14. Untuk mengonfigurasi enkripsi default, di bawah Jenis enkripsi, pilih salah satu dari berikut ini:

    - Kunci yang dikelola Amazon S3 (SSE-S3)
    - AWS Key Management Service kunci (SSE-KMS)

    > **Important**
    >
    > Jika Anda menggunakan opsi SSE-KMS untuk konfigurasi enkripsi default, Anda tunduk pada kuota permintaan per detik (RPS) AWS KMS. Untuk informasi selengkapnya tentang AWS KMS kuota dan cara meminta kenaikan kuota, lihat Kuota di Panduan Pengembang AWS Key Management Service .

    Bucket dan objek baru dienkripsi dengan enkripsi di sisi server dengan kunci yang dikelola Amazon S3 sebagai tingkat dasar konfigurasi enkripsi. Untuk informasi selengkapnya tentang enkripsi default, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3.

Untuk informasi selengkapnya tentang penggunaan enkripsi di sisi server Amazon S3 guna mengenkripsi data Anda, lihat Menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3).

15. Jika Anda memilih kunci AWS Key Management Service (SSE-KMS), lakukan hal berikut:

    a. Di bawah AWS KMS kunci, tentukan kunci KMS Anda dengan salah satu cara berikut ini:

    - Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari Anda AWS KMS keys, dan pilih kunci KMS Anda dari daftar kunci yang tersedia.

      Kunci Kunci yang dikelola AWS (`aws/s3`) dan kunci terkelola pelanggan Anda muncul dalam daftar ini. Untuk informasi selengkapnya tentang CMK, lihat Kunci pelanggan dan AWS kunci di AWS Key Management Service Panduan Pengembang.
    - Untuk memasukkan ARN kunci KMS, pilih Masukkan AWS KMS key ARN, dan masukkan ARN kunci KMS Anda di bidang yang muncul.
    - Untuk membuat kunci terkelola pelanggan baru di AWS KMS konsol, pilih Buat kunci KMS.

      Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat kunci di Panduan AWS Key Management Service Pengembang.

> **Important**
>
> Anda hanya dapat menggunakan tombol KMS yang tersedia Wilayah AWS sama dengan bucket. Konsol Amazon S3 hanya mencantumkan kunci 100 KMS pertama di Wilayah yang sama dengan bucket. Untuk menggunakan kunci KMS yang tidak terdaftar, Anda harus memasukkan ARN kunci KMS Anda. Jika Anda ingin menggunakan kunci KMS yang dimiliki oleh akun lain, Anda harus terlebih dahulu memiliki izin untuk menggunakan kunci tersebut, dan kemudian Anda harus memasukkan ARN kunci KMS. Untuk informasi selengkapnya tentang izin lintas akun untuk kunci KMS, lihat Membuat kunci KMS yang dapat digunakan akun lain di Panduan Pengembang AWS Key Management Service . Untuk informasi selengkapnya tentang SSE-KMS, lihat Menentukan enkripsi di sisi server dengan AWS KMS (SSE-KMS).
>
> Saat Anda menggunakan enkripsi sisi server AWS KMS key untuk Amazon S3, Anda harus memilih kunci KMS enkripsi simetris. Amazon S3 hanya mendukung

kunci KMS enkripsi simetris dan tidak mendukung kunci KMS asimetris. Untuk informasi selengkapnya, lihat Mengidentifikasi tombol KMS simetris dan asimetris dalam Panduan Pengembang AWS Key Management Service .

Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat kunci di Panduan AWS Key Management Service Pengembang. Untuk informasi selengkapnya tentang penggunaan AWS KMS dengan Amazon S3, lihat. Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS)

b. Saat mengonfigurasi bucket untuk menggunakan enkripsi default dengan SSE-KMS, Anda juga dapat mengaktifkan Kunci Bucket S3. S3 Bucket Keys menurunkan biaya enkripsi dengan mengurangi lalu lintas permintaan dari Amazon S3 ke. AWS KMS Untuk informasi selengkapnya, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3.

   Untuk menggunakan Kunci Bucket S3, di bagian bawah Kunci Bucket, pilih Aktifkan.

16. (Opsional) Jika Anda ingin mengaktifkan Kunci Objek S3, lakukan hal berikut ini:

    a. Pilih Pengaturan lanjutan.

    > **Important**
    >
    > Mengaktifkan Kunci Objek juga mengaktifkan Penentuan Versi untuk bucket. Setelah mengaktifkan, Anda harus mengonfigurasi pengaturan penyimpanan default Kunci Objek dan penahanan legal untuk melindungi objek baru agar tidak dihapus atau ditimpa.

    b. Jika Anda ingin mengaktifkan Kunci Objek, pilih Aktifkan, baca peringatan yang muncul, lalu setujui.

    Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek S3.

    > **Note**
    >
    > Untuk membuat bucket dengan dukungan Kunci Objek, Anda harus memiliki izin berikut: `s3:CreateBucket`, `s3:PutBucketVersioning` dan `s3:PutBucketObjectLockConfiguration`.

17. Pilih Buat bucket.

Anda telah membuat bucket di Amazon S3.

Langkah selanjutnya

Untuk menambahkan objek ke bucket Anda, lihat Langkah 2: Unggah satu objek ke bucket Anda.

Langkah 2: Unggah satu objek ke bucket Anda

Setelah membuat bucket di Amazon S3, Anda siap untuk mengunggah objek ke bucket. Objek dapat berupa jenis file apa pun: file teks, foto, video, dan sebagainya.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Untuk mengunggah objek ke bucket

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket tempat Anda ingin mengunggah objek.
3. Di tab Objek untuk bucket Anda, pilih Unggah.
4. Di Bawah File dan folder, pilih Tambahkan file.
5. Pilih file yang akan diunggah, lalu pilih Buka.
6. Pilih Unggah.

Anda berhasil mengunggah objek ke bucket Anda.

Langkah selanjutnya

Untuk melihat objek Anda, lihat Langkah 3: Mengunduh objek.

Langkah 3: Mengunduh objek

Setelah mengunggah sebuah objek ke bucket, Anda dapat melihat informasi tentang objek tersebut dan mengunduhnya ke komputer lokal.

> **Note**
>
> Untuk informasi selengkapnya tentang penggunaan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Menggunakan konsol S3

Bagian ini menjelaskan cara menggunakan konsol Amazon S3 untuk mengunduh objek dari bucket S3.

> **Note**
>
> - Anda hanya dapat mengunduh satu objek dalam satu waktu.
> - Jika Anda menggunakan konsol Amazon S3 untuk mengunduh objek yang nama kuncinya diakhiri dengan titik (`.`), titik tersebut dihapus dari nama kunci objek yang diunduh. Untuk mempertahankan periode di akhir nama objek yang diunduh, Anda harus menggunakan AWS Command Line Interface (AWS CLI), SDK AWS, atau API REST Amazon S3.

Untuk mengunduh objek dari bucket S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Bucket, pilih nama bucket yang ingin Anda unduh objeknya.
3. Anda dapat mengunduh objek dari bucket S3 dengan cara berikut:
   - Pilih kotak centang di samping objek, dan pilih Unduh. Jika Anda ingin mengunduh objek ke folder tertentu, pada menu Tindakan, pilih Unduh sebagai.
   - Jika Anda ingin mengunduh versi objek tertentu, aktifkan Tampilkan versi (terletak di samping kotak pencarian). Centang kotak di samping versi objek yang Anda inginkan, dan pilih Unduh.

Jika Anda ingin mengunduh objek ke folder tertentu, pada menu Tindakan, pilih Unduh sebagai.

Anda berhasil mengunduh objek.

Langkah selanjutnya

Untuk menyalin dan menempel objek Anda di Amazon S3, lihat Langkah 4: Salin objek ke folder.

Langkah 4: Salin objek ke folder

Anda sudah menambahkan objek ke dalam bucket dan mengunduh objek. Sekarang, buat folder dan salin objek dan tempelkan ke folder tersebut.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Untuk menyalin objek ke folder

1. Di daftar Bucket, pilih nama bucket Anda.
2. Pilih Buat folder dan konfigurasikan folder baru:
   a. Masukkan nama folder (misalnya, `favorite-pics`).
   b. Untuk pengaturan enkripsi folder, pilih Nonaktifkan.
   c. Pilih Simpan.
3. Navigasikan ke bucket atau folder Amazon S3 yang berisi objek yang ingin Anda salin.
4. Centang kotak di sebelah kiri nama objek yang ingin Anda salin.
5. Pilih Tindakan dan pilih Salin dari daftar opsi yang muncul.

   Atau, pilih Salin dari opsi di kanan atas.
6. Pilih folder tujuan:
   a. Pilih Jelajahi S3.

b. Pilih tombol opsi di sebelah kiri nama folder.

   Untuk menavigasiakn ke folder dan memilih subfolder sebagai tujuan, pilih nama folder.

c. Pilih Pilih tujuan.

Jalur ke folder tujuan akan muncul di kotak Tujuan. Di Tujuan, Anda dapat memasukkan jalur tujuan Anda secara bergantian, misalnya, s3://*bucket-name*/*folder-name*/.

7. Di kanan bawah, pilih Salin.

   Amazon S3 menyalin objek Anda ke folder tujuan.

Langkah selanjutnya

Untuk menghapus objek dan bucket di Amazon S3, lihat Langkah 5: Hapus objek dan bucket Anda.

Langkah 5: Hapus objek dan bucket Anda

Saat Anda tidak lagi memerlukan objek atau bucket, kami menyarankan Anda menghapusnya untuk mencegah biaya lebih lanjut. Jika Anda menyelesaikan panduan memulai ini sebagai latihan pembelajaran, dan Anda tidak berencana untuk menggunakan bucket atau objek Anda, kami sarankan Anda menghapus bucket dan objek Anda sehingga biaya tidak lagi bertambah.

Sebelum menghapus bucket, kosongkan bucket atau hapus objek di dalam bucket. Setelah Anda menghapus objek dan bucket Anda, objek tersebut tidak lagi tersedia.

Jika Anda ingin terus menggunakan nama bucket yang sama, sebaiknya hapus objek atau kosongkan bucket, tetapi jangan menghapus bucket. Setelah Anda menghapus bucket, nama akan tersedia untuk digunakan kembali. Namun, Akun AWS lain mungkin membuat bucket dengan nama yang sama sebelum Anda sempat menggunakannya kembali.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan direktori bucket, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Topik

- Menghapus objek
- Mengosongkan bucket Anda
- Menghapus bucket Anda

Menghapus objek

Jika ingin memilih objek mana yang Anda hapus tanpa mengosongkan semua objek dari bucket, Anda dapat menghapus objek.

1. Di daftar Bucket, pilih nama bucket yang ingin Anda hapus objeknya.
2. Pilih objek yang ingin Anda hapus.
3. Pilih Hapus dari opsi di kanan atas.
4. Pada halaman Hapus objek, ketik **`delete`** untuk mengonfirmasi penghapusan objek Anda.
5. Pilih Hapus objek.

Mengosongkan bucket Anda

Apabila Anda berencana menghapus bucket, Anda harus terlebih dahulu mengosongkan bucket Anda, yang akan menghapus seluruh objek dalam bucket.

Mengosongkan bucket

1. Di daftar Bucket, pilih bucket yang ingin Anda kosongkan, lalu pilih Kosongkan.
2. Untuk mengonfirmasi bahwa Anda ingin mengosongkan bucket dan menghapus semua objek di dalamnya, di Bucket kosong, ketik **`permanently delete`**.

   > **Important**
   >
   > Mengosongkan bucket tidak dapat diurungkan. Objek yang ditambahkan ke bucket saat tindakan bucket kosong sedang berlangsung akan dihapus.

3. Untuk mengosongkan bucket dan menghapus semua objek di dalamnya, lalu pilih Kosongkan.

   Halaman Bucket kosong: Status yang terbuka dapat Anda gunakan untuk meninjau ringkasan penghapusan objek yang gagal dan berhasil.

4. Untuk kembali ke daftar bucket Anda, pilih Keluar.

Menghapus bucket Anda

Setelah mengosongkan bucket atau menghapus semua objek dari bucket, Anda dapat menghapus bucket.

1. Untuk menghapus bucket, dalam daftar Bucket, pilih bucket.
2. Pilih Hapus.
3. Untuk mengonfirmasi penghapusan, di Hapus bucket, ketik nama bucket.

> **Important**
>
> Menghapus bucket tidak dapat dibatalkan. Nama bucket bersifat unik. Jika Anda menghapus bucket, pengguna AWS lain dapat menggunakan nama tersebut. Jika Anda ingin terus menggunakan nama bucket yang sama, jangan menghapus bucket Anda. Sebaliknya, kosongkan dan simpan bucket.

4. Untuk menghapus bucket Anda, pilih Hapus bucket.

Langkah selanjutnya

Dalam contoh sebelumnya, Anda telah mempelajari cara melakukan beberapa tugas dasar Amazon S3.

Topik-topik berikut ini menjelaskan jalur pembelajaran yang dapat Anda gunakan untuk mendapatkan pemahaman yang lebih mendalam tentang Amazon S3 sehingga Anda dapat mengimplementasikannya di aplikasi Anda.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Topik

- Memahami kasus penggunaan umum
- Mengontrol akses ke bucket dan objek Anda
- Mengelola dan memantau penyimpanan Anda
- Mengembangkan dengan Amazon S3
- Belajar dari tutorial
- Menjelajahi pelatihan dan dukungan

Memahami kasus penggunaan umum

Anda dapat menggunakan Amazon S3 untuk mendukung kasus penggunaan spesifik Anda. AWS Pustaka Solusi dan AWS Blog menyediakan informasi dan tutorial khusus kasus penggunaan. Berikut ini adalah beberapa kasus penggunaan umum untuk Amazon S3:

- Cadangan dan penyimpanan–Menggunakan fitur manajemen penyimpanan Amazon S3 untuk mengelola biaya, memenuhi persyaratan peraturan, mengurangi latensi, dan menyimpan beberapa salinan data Anda yang berbeda untuk persyaratan kepatuhan.
- Hosting aplikasi–Menyebarkan, menginstal, dan mengelola aplikasi web yang andal, sangat dapat diskalakan, dan berbiaya rendah. Misalnya, Anda dapat mengonfigurasi bucket Amazon S3 untuk melakukan hosting situs web statis. Untuk informasi selengkapnya, lihat Hosting situs web statis menggunakan Amazon S3.
- Hosting media–Membangun infrastruktur yang sangat tersedia yang menampung unggahan dan unduhan video, foto, atau musik.
- Pengiriman perangkat lunak–Melakukan hosting aplikasi perangkat lunak untuk diunduh pelanggan.

Mengontrol akses ke bucket dan objek Anda

Amazon S3 menyediakan beragam fitur dan alat keamanan. Untuk ikhtisar, lihat Manajemen Akses.

Secara default, bucket S3 dan objek di dalamnya bersifat pribadi. Anda hanya memiliki akses ke sumber daya S3 yang Anda buat. Anda dapat menggunakan fitur berikut untuk memberikan izin sumber daya terperinci yang mendukung kasus penggunaan spesifik Anda atau untuk mengaudit izin sumber daya Amazon S3 Anda.

- Blokir Akses Publik S3–Memblokir akses publik ke bucket dan objek S3. Secara default, pengaturan Blokir Akses Publik diaktifkan di tingkat bucket.

- AWS Identity and Access Management Identitas (IAM) — Gunakan IAM atau AWS IAM Identity Center untuk membuat identitas IAM di Anda untuk mengelola akses Akun AWS ke sumber daya Amazon S3 Anda. Misalnya, Anda dapat menggunakan IAM dengan Amazon S3 untuk mengontrol jenis akses yang dimiliki pengguna atau grup pengguna ke bucket Amazon S3 yang Anda miliki. Akun AWS Untuk informasi selengkapnya tentang identitas IAM dan praktik terbaik, lihat Identitas IAM (pengguna, grup pengguna, dan peran) dalam Panduan Pengguna IAM.
- Kebijakan bucket–Gunakan bahasa kebijakan berbasis IAM untuk mengonfigurasi izin berbasis sumber daya untuk bucket S3 dan objek di dalamnya.
- Daftar kontrol akses (ACL)–Berikan izin baca dan tulis untuk masing-masing bucket dan objek kepada pengguna yang berwenang. Sebagai aturan umum, sebaiknya gunakan kebijakan berbasis sumber daya S3 (kebijakan bucket dan kebijakan titik akses) atau kebijakan pengguna IAM untuk kontrol akses, bukan ACL. Kebijakan adalah opsi kontrol akses yang disederhanakan dan lebih fleksibel. Dengan kebijakan bucket dan kebijakan titik akses, Anda dapat menentukan aturan yang berlaku secara luas di semua permintaan ke sumber daya Amazon S3 Anda. Untuk informasi selengkapnya tentang kasus spesifik saat Anda menggunakan ACL, bukan kebijakan berbasis sumber daya atau kebijakan pengguna IAM, lihat. Identity and Access Management untuk Amazon S3
- Kepemilikan Objek S3–Ambil kepemilikan setiap objek di bucket Anda, sederhanakan manajemen akses untuk data yang disimpan di Amazon S3. Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk menonaktifkan atau mengaktifkan ACL. Secara default, ACL dinonaktifkan. Ketika ACL dinonaktifkan, pemilik bucket memiliki semua objek di bucket dan mengelola akses ke data secara eksklusif dengan menggunakan kebijakan manajemen akses.
- IAM Access Analyzer for S3–Mengevaluasi dan memantau kebijakan akses bucket S3 Anda, memastikan bahwa kebijakan hanya menyediakan akses yang dimaksudkan ke sumber daya S3 Anda.

Mengelola dan memantau penyimpanan Anda

- Mengelola penyimpanan–Setelah Anda membuat bucket dan mengunggah objek di Amazon S3, Anda dapat mengelola penyimpanan objek Anda. Misalnya, Anda dapat menggunakan Penentuan Versi S3 dan Replikasi S3 untuk pemulihan bencana, Siklus Hidup S3 untuk mengelola biaya penyimpanan, dan Kunci Objek S3 untuk memenuhi persyaratan kepatuhan.
- Memantau penyimpanan Anda — Pemantauan adalah bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja Amazon S3 dan solusi Anda AWS . Anda dapat memantau aktivitas dan

biaya penyimpanan. Selain itu, kami menyarankan Anda mengumpulkan data pemantauan dari semua bagian solusi AWS Anda sehingga Anda dapat lebih mudah melakukan debug kegagalan multipoin jika terjadi.

- Analitik dan wawasan–Anda juga dapat menggunakan analitik dan wawasan di Amazon S3 untuk memahami, menganalisis, dan mengoptimalkan penggunaan penyimpanan Anda. Misalnya, menggunakan Lensa Penyimpanan Amazon S3 untuk memahami, menganalisis, dan mengoptimalkan penyimpanan Anda. Lensa Penyimpanan S3 menyediakan 29+ metrik penggunaan dan aktivitas serta dasbor interaktif untuk mengumpulkan data untuk seluruh organisasi, akun tertentu, Wilayah, bucket, atau awalan Anda. Gunakan Analisis Kelas Penyimpanan untuk menganalisis pola akses penyimpanan guna memutuskan kapan saatnya memindahkan data Anda ke kelas penyimpanan yang lebih hemat biaya.

Mengembangkan dengan Amazon S3

Amazon S3 adalah layanan REST. Anda dapat mengirim permintaan ke Amazon S3 menggunakan REST API atau pustaka AWS SDK, yang membungkus API Amazon S3 REST yang mendasarinya, menyederhanakan tugas pemrograman Anda. Anda juga dapat menggunakan AWS Command Line Interface (AWS CLI) untuk melakukan panggilan API Amazon S3. Untuk informasi selengkapnya, lihat Membuat permintaan.

API REST Amazon S3 adalah antarmuka HTTP ke Amazon S3. Dengan API REST, Anda menggunakan permintaan HTTP standar untuk membuat, dan menghapus bucket dan objek. Untuk menggunakan API REST, Anda dapat menggunakan toolkit yang mendukung HTTP. Anda bahkan dapat menggunakan peramban untuk mengambil objek, selama objek tersebut dapat dibaca secara anonim. Untuk informasi selengkapnya, lihat Berkembang dengan Amazon S3 menggunakan API REST.

Untuk membantu Anda membangun aplikasi menggunakan bahasa pilihan Anda, kami menyediakan sumber daya berikut.

AWS CLI

Anda dapat mengakses fitur-fitur Amazon S3 menggunakan file AWS CLI. Untuk mengunduh dan mengkonfigurasi AWS CLI, lihatMengembangkan dengan Amazon S3 menggunakan AWS CLI.

AWS CLI Ini menyediakan dua tingkatan perintah untuk mengakses Amazon S3: perintah tingkat tinggi (s3) dan perintah tingkat API (s3api dan s3control). Perintah S3 tingkat tinggi menyederhanakan operasi tugas-tugas umum, seperti membuat, memanipulasi, dan menghapus

objek dan bucket. Perintah s3api dan s3control mengekspos akses langsung ke semua operasi Amazon S3 API, yang dapat Anda gunakan untuk melakukan operasi lanjutan yang mungkin tidak mungkin dilakukan dengan perintah tingkat tinggi saja.

Untuk daftar AWS CLI perintah Amazon S3, lihat s3, s3api, dan s3control.

AWS SDK dan Penjelajah

Anda dapat menggunakan AWS SDK saat mengembangkan aplikasi dengan Amazon S3. SDK AWS menyederhanakan tugas pemrograman Anda dengan membungkus API REST yang mendasarinya. AWS Mobile SDK dan JavaScript Amplify library juga tersedia untuk membangun aplikasi seluler dan web yang terhubung. AWS

Selain AWS SDK, AWS Explorers tersedia untuk Visual Studio dan Eclipse untuk Java IDE. Dalam hal ini, SDK dan penjelajah dibundel bersama sebagai Toolkit. AWS

Untuk informasi selengkapnya, lihat Mengembangkan dengan Amazon S3 menggunakan SDK AWS.

Sampel Kode dan Pustaka

AWS Pusat Developer dan AWS Katalog Sampel Kode memiliki sampel kode dan pustaka yang ditulis khusus untuk Amazon S3. Anda dapat menggunakan sampel kode ini untuk memahami cara mengimplementasikan API Amazon S3. Anda juga dapat melihat Referensi API Amazon Simple Storage Service untuk memahami operasi Amazon S3 API secara detail.

Belajar dari tutorial

Anda dapat memulai dengan step-by-step tutorial untuk mempelajari lebih lanjut tentang Amazon S3. Tutorial ini ditujukan untuk lingkungan tipe lab, dan tutorial menggunakan nama perusahaan fiktif, nama pengguna fikif, dan sebagainya. Tujuannya adalah untuk memberikan pedoman umum. Produk ini tidak dimaksudkan untuk penggunaan secara langsung di lingkungan produksi tanpa tinjauan dan adaptasi yang cermat untuk memenuhi kebutuhan unik di lingkungan organisasi Anda.

Memulai

- Tutorial: Menyimpan dan mengambil file dengan Amazon S3
- Tutorial: Mulai menggunakan S3 Intelligent-Tiering
- Tutorial: Mulai menggunakan kelas penyimpanan Amazon S3 Glacier

Mengoptimalkan biaya penyimpanan

- Tutorial: Mulai menggunakan S3 Intelligent-Tiering
- Tutorial: Mulai menggunakan kelas penyimpanan Amazon S3 Glacier
- Tutorial: Mengoptimalkan biaya dan mendapatkan visibilitas ke dalam penggunaan dengan Lensa Penyimpanan S3

Mengelola penyimpanan

- Tutorial: Memulai dengan Titik Akses Multi-Wilayah Amazon S3
- Tutorial: Mereplikasi objek yang ada di bucket Amazon S3 Anda dengan Replikasi Batch S3

Hosting video dan situs web

- Tutorial: Hosting video streaming sesuai permintaan dengan Amazon S3, Amazon, dan CloudFront Amazon Route 53
- Tutorial: Mengonfigurasi situs web statis untuk Amazon S3
- Tutorial: Mengonfigurasi situs web statis menggunakan domain kustom yang terdaftar di Route 53

Pemrosesan data

- Tutorial: Mengubah data untuk aplikasi Anda dengan S3 Object Lambda
- Tutorial: Mendeteksi dan menyunting data PII dengan S3 Object Lambda dan Amazon Comprehend
- Tutorial: Menggunakan S3 Lambda Objek untuk menandai gambar secara dinamis saat gambarnya diambil
- Tutorial: Video transcoding batch dengan Operasi Batch S3,, dan AWS LambdaAWS Elemental MediaConvert

Melindungi data

- Tutorial: Memeriksa integritas data di Amazon S3 dengan checksum tambahan
- Tutorial: Mereplikasi data di dalam dan di antara Wilayah AWS menggunakan Replikasi S3

- Tutorial: Melindungi data di Amazon S3 dari penghapusan yang tidak disengaja atau bug aplikasi menggunakan Penentuan Versi S3, Kunci Objek S3, dan Replikasi S3
- Tutorial: Mereplikasi objek yang ada di bucket Amazon S3 Anda dengan Replikasi Batch S3

Menjelajahi pelatihan dan dukungan

Anda dapat belajar dari AWS para ahli untuk memajukan keterampilan Anda dan mendapatkan bantuan ahli untuk mencapai tujuan Anda.

- Pelatihan–Materi pelatihan memberikan pendekatan langsung untuk mempelajari Amazon S3. Untuk informasi selengkapnya, lihat AWS Pelatihan dan sertifikasi dan AWS Pembicaraan teknologi online.
- Forum Diskusi–Di forum, Anda dapat meninjau posting untuk memahami apa yang dapat dan tidak dapat Anda lakukan dengan Amazon S3. Anda juga dapat memposting pertanyaan Anda. Untuk informasi selengkapnya, lihat Forum Diskusi.
- Dukungan Teknis–Jika Anda memiliki pertanyaan lebih lanjut, Anda dapat menghubungi Dukungan Teknis.

Tutorial

Tutorial berikut menyajikan end-to-end prosedur lengkap untuk tugas-tugas Amazon S3 umum. Tutorial ini ditujukan untuk lingkungan tipe lab, dan tutorial menggunakan nama perusahaan fiktif, nama pengguna fikif, dan sebagainya. Tujuannya adalah untuk memberikan pedoman umum. Produk ini tidak dimaksudkan untuk penggunaan secara langsung di lingkungan produksi tanpa tinjauan dan adaptasi yang cermat untuk memenuhi kebutuhan unik di lingkungan organisasi Anda.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Memulai

- Tutorial: Menyimpan dan mengambil file dengan Amazon S3
- Tutorial: Mulai menggunakan S3 Intelligent-Tiering
- Tutorial: Mulai menggunakan kelas penyimpanan Amazon S3 Glacier

Mengoptimalkan biaya penyimpanan

- Tutorial: Mulai menggunakan S3 Intelligent-Tiering
- Tutorial: Mulai menggunakan kelas penyimpanan Amazon S3 Glacier
- Tutorial: Mengoptimalkan biaya dan mendapatkan visibilitas ke dalam penggunaan dengan Lensa Penyimpanan S3

Mengelola penyimpanan

- Tutorial: Memulai dengan Titik Akses Multi-Wilayah Amazon S3
- Tutorial: Mereplikasi objek yang ada di bucket Amazon S3 Anda dengan Replikasi Batch S3

Hosting video dan situs web

- Tutorial: Hosting video streaming sesuai permintaan dengan Amazon S3, Amazon, dan CloudFront Amazon Route 53
- Tutorial: Mengonfigurasi situs web statis untuk Amazon S3
- Tutorial: Mengonfigurasi situs web statis menggunakan domain kustom yang terdaftar di Route 53

Pemrosesan data

- Tutorial: Mengubah data untuk aplikasi Anda dengan S3 Object Lambda
- Tutorial: Mendeteksi dan menyunting data PII dengan S3 Object Lambda dan Amazon Comprehend
- Tutorial: Menggunakan S3 Lambda Objek untuk menandai gambar secara dinamis saat gambarnya diambil
- Tutorial: Video transcoding batch dengan Operasi Batch S3,, dan AWS LambdaAWS Elemental MediaConvert

Melindungi data

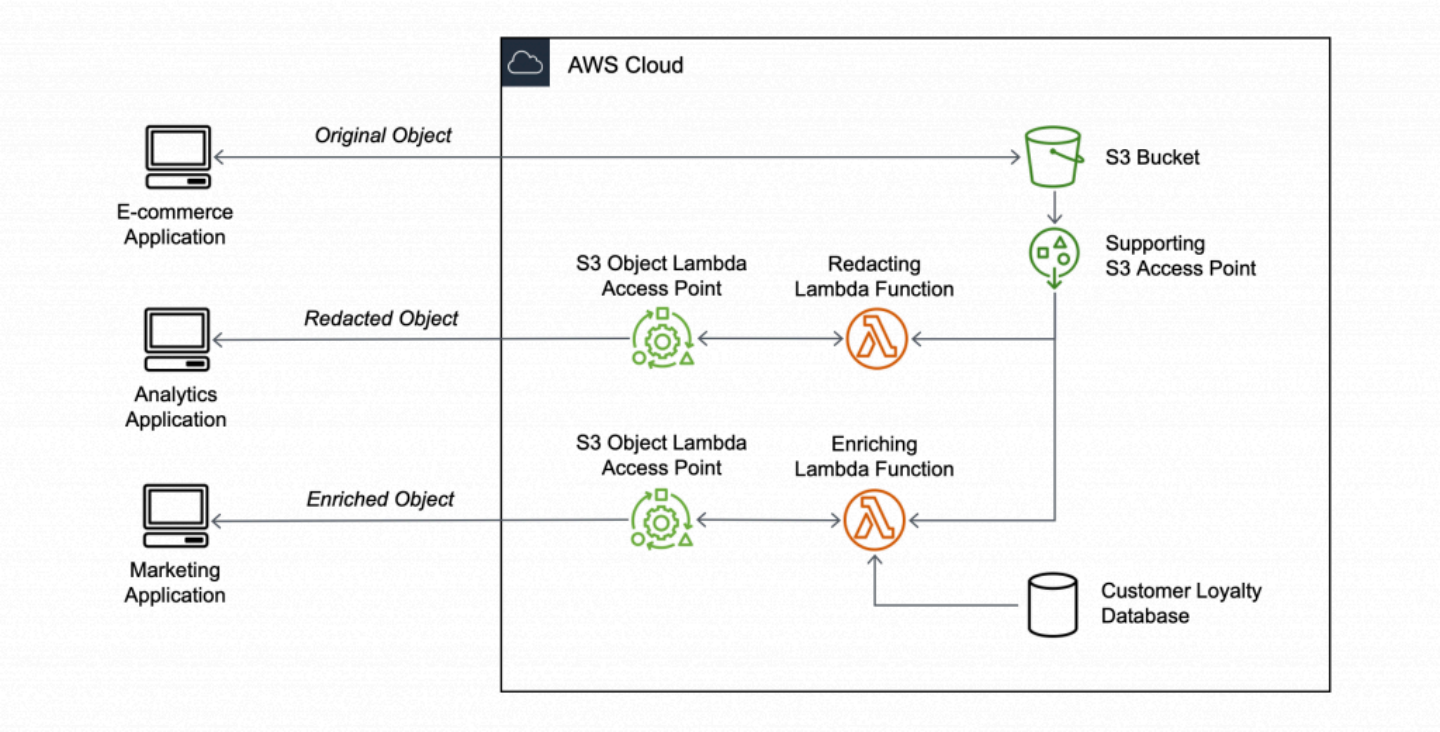
- Tutorial: Memeriksa integritas data di Amazon S3 dengan checksum tambahan
- Tutorial: Mereplikasi data di dalam dan di antara Wilayah AWS menggunakan Replikasi S3
- Tutorial: Melindungi data di Amazon S3 dari penghapusan yang tidak disengaja atau bug aplikasi menggunakan Penentuan Versi S3, Kunci Objek S3, dan Replikasi S3
- Tutorial: Mereplikasi objek yang ada di bucket Amazon S3 Anda dengan Replikasi Batch S3

Tutorial: Mengubah data untuk aplikasi Anda dengan S3 Object Lambda

Saat Anda menyimpan data di Amazon S3, Anda dapat dengan mudah membagikannya untuk digunakan oleh beberapa aplikasi. Namun, setiap aplikasi mungkin memiliki persyaratan format data yang unik, dan mungkin memerlukan modifikasi atau pemrosesan data Anda untuk kasus penggunaan tertentu. Misalnya, kumpulan data yang dibuat oleh aplikasi e-niaga mungkin termasuk informasi identitas pribadi (PII). Ketika data yang sama diproses untuk analitik, PII ini tidak diperlukan

dan harus disunting. Namun, jika dataset yang sama digunakan untuk kampanye pemasaran, Anda mungkin perlu memperkaya data dengan detail tambahan, seperti informasi dari database loyalitas pelanggan.

Dengan S3 Object Lambda, Anda dapat menambahkan kode Anda sendiri untuk memproses data yang diambil dari S3 sebelum mengembalikannya ke aplikasi. Secara khusus, Anda dapat mengonfigurasi AWS Lambda fungsi dan melampirkannya ke Titik Akses Lambda Objek S3. Saat aplikasi mengirimkan permintaan GET S3 standar melalui Titik Akses Lambda Objek S3, fungsi Lambda yang ditentukan dipanggil untuk memproses data apa pun yang diambil dari bucket S3 melalui jalur akses S3 pendukung. Kemudian, Titik Akses Lambda Objek S3 mengembalikan hasil yang diubah kembali ke aplikasi. Anda dapat membuat dan menjalankan fungsi Lambda kustom Anda sendiri, menyesuaikan transformasi data Lambda Objek S3 dengan kasus penggunaan spesifik Anda, semua tanpa perubahan yang diperlukan untuk aplikasi Anda.



Tujuan

Dalam tutorial ini, Anda mempelajari cara menambahkan kode kustom ke permintaan GET S3 standar untuk memodifikasi objek yang diminta diambil dari S3 sehingga objek sesuai dengan kebutuhan klien atau aplikasi yang meminta. Secara khusus, Anda mempelajari cara mengubah semua teks dalam objek asli yang disimpan di S3 menjadi huruf besar melalui S3 Object Lambda.

Topik

- Prasyarat
- Langkah 1: Buat ember S3
- Langkah 2: Unggah file ke bucket S3
- Langkah 3: Buat titik akses S3
- Langkah 4: Buat fungsi Lambda
- Langkah 5: Konfigurasikan kebijakan IAM untuk peran eksekusi fungsi Lambda Anda
- Langkah 6: Buat Titik Akses Lambda Objek S3
- Langkah 7: Lihat data yang diubah
- Langkah 8: Membersihkan
- Langkah selanjutnya

Prasyarat

Sebelum Anda memulai tutorial ini, Anda harus memiliki Akun AWS yang dapat Anda masuki sebagai pengguna AWS Identity and Access Management (IAM) dengan izin yang benar. Anda juga harus menginstal Python versi 3.8 atau yang lebih baru.

Sublangkah

- Buat pengguna IAM dengan izin di Akun AWS (konsol)
- Instal Python 3.8 atau yang lebih baru di komputer lokal Anda

Buat pengguna IAM dengan izin di Akun AWS (konsol)

Anda dapat membuat pengguna IAM untuk tutorial. Untuk menyelesaikan tutorial ini, pengguna IAM Anda harus melampirkan kebijakan IAM berikut untuk mengakses AWS sumber daya yang relevan dan melakukan tindakan tertentu. Untuk informasi selengkapnya tentang cara membuat pengguna IAM, lihat Membuat pengguna IAM (konsol) di Panduan Pengguna IAM.

Pengguna IAM Anda memerlukan kebijakan berikut:

- AmazonS3 FullAccess - Memberikan izin untuk semua tindakan Amazon S3, termasuk izin untuk membuat dan menggunakan Titik Akses Lambda Objek.
- AWSLambda_FullAccess— Memberikan izin untuk semua tindakan Lambda.
- IAM FullAccess — Memberikan izin untuk semua tindakan IAM.

- IAM AccessAnalyzerReadOnlyAccess — Memberikan izin untuk membaca semua informasi akses yang disediakan oleh IAM Access Analyzer.
- CloudWatchLogsFullAccess— Memberikan akses penuh ke CloudWatch Log.

> **Note**
>
> Untuk kesederhanaan, tutorial ini membuat dan menggunakan pengguna IAM. Setelah menyelesaikan tutorial ini, ingatlah untukHapus pengguna IAM. Untuk penggunaan produksi, kami menyarankan Anda mengikuti praktik terbaik Keamanan di IAM dalam Panduan Pengguna IAM. Praktik terbaik mengharuskan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS dengan kredensi sementara. Praktik terbaik lainnya adalah meminta beban kerja untuk menggunakan kredensil sementara dengan peran IAM untuk mengakses. AWS Untuk mempelajari cara menggunakan AWS IAM Identity Center untuk membuat pengguna dengan kredensyal sementara, lihat Memulai di AWS IAM Identity Center Panduan Pengguna.
> Tutorial ini juga menggunakan kebijakan AWS terkelola akses penuh. Untuk penggunaan produksi, sebaiknya Anda hanya memberikan izin minimum yang diperlukan untuk kasus penggunaan Anda, sesuai dengan praktik terbaik keamanan.

Instal Python 3.8 atau yang lebih baru di komputer lokal Anda

Gunakan prosedur berikut untuk menginstal Python 3.8 atau yang lebih baru pada mesin lokal Anda. Untuk petunjuk penginstalan lainnya, lihat halaman Mengunduh Python di Panduan Pemula Python.

1. Buka terminal atau shell lokal Anda dan jalankan perintah berikut untuk menentukan apakah Python sudah diinstal, dan jika demikian, versi mana yang diinstal.

   ```
   python --version
   ```

2. Jika Anda tidak memiliki Python 3.8 atau yang lebih baru, unduh penginstal resmi Python 3.8 atau yang lebih baru yang cocok untuk mesin lokal Anda.
3. Jalankan penginstal dengan mengklik dua kali file yang diunduh, dan ikuti langkah-langkah untuk menyelesaikan instalasi.

   Untuk pengguna Windows, pilih Tambahkan Python 3.X ke PATH di wizard instalasi sebelum memilih Instal Sekarang.

4. Mulai ulang terminal Anda dengan menutup dan membukanya kembali.
5. Jalankan perintah berikut untuk memverifikasi bahwa Python 3.8 atau yang lebih baru diinstal dengan benar.

   Untuk pengguna macOS, jalankan perintah ini:

   ```
   python3 --version
   ```

   Untuk pengguna Windows, jalankan perintah ini:

   ```
   python --version
   ```

6. Jalankan perintah berikut untuk memverifikasi bahwa manajer paket pip3 diinstal. Jika Anda melihat nomor versi pip dan python 3.8 atau yang lebih baru dalam respons perintah, itu berarti manajer paket pip3 berhasil diinstal.

   ```
   pip --version
   ```

Langkah 1: Buat ember S3

Buat bucket untuk menyimpan data asli yang Anda rencanakan untuk diubah.

Untuk membuat bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih Buat bucket.

   Halaman Buat bucket terbuka.
4. Untuk nama Bucket, masukkan nama (misalnya,**`tutorial-bucket`**) untuk bucket Anda.

   Untuk informasi selengkapnya tentang penamaan bucket di Amazon S3, lihat. Peraturan penamaan bucket
5. Untuk Wilayah, pilih Wilayah AWS tempat Anda ingin ember berada.

   Untuk informasi selengkapnya tentang Wilayah bucket, lihatGambaran umum bucket.

6. Untuk pengaturan Blokir Akses Publik untuk bucket ini, pertahankan pengaturan default (Blokir semua akses publik diaktifkan).

   Kami menyarankan agar Anda tetap mengaktifkan semua pengaturan Blokir Akses Publik kecuali Anda perlu menonaktifkan satu atau beberapa pengaturan tersebut untuk kasus penggunaan Anda. Untuk informasi lebih lanjut tentang pemblokiran akses publik, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.
7. Untuk pengaturan yang tersisa, pertahankan defaultnya.

   (Opsional) Jika Anda ingin mengonfigurasi pengaturan bucket tambahan untuk kasus penggunaan spesifik Anda, lihatMembuat bucket.
8. Pilih Buat bucket.

Langkah 2: Unggah file ke bucket S3

Unggah file teks ke bucket S3. File teks ini berisi data asli yang akan Anda ubah menjadi huruf besar nanti dalam tutorial ini.

Misalnya, Anda dapat mengunggah `tutorial.txt` file yang berisi teks berikut:

```
Amazon S3 Object Lambda Tutorial:
You can add your own code to process data retrieved from S3 before
returning it to an application.
```

Untuk mengunggah file ke ember

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dalam daftar Bucket, pilih nama bucket yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**) untuk mengunggah file Anda.
4. Di tab Objek untuk bucket Anda, pilih Unggah.
5. Pada halaman Unggah, di bawah File dan folder, pilih Tambahkan file.
6. Pilih file yang akan diunggah, lalu pilih Buka. Misalnya, Anda dapat mengunggah contoh `tutorial.txt` file yang disebutkan sebelumnya.
7. Pilih Unggah.

Langkah 3: Buat titik akses S3

Untuk menggunakan Titik Akses Lambda Objek S3 untuk mengakses dan mengubah data asli, Anda harus membuat titik akses S3 dan mengaitkannya dengan bucket S3 yang Anda buat di Langkah 1. Titik akses harus Wilayah AWS sama dengan objek yang ingin Anda ubah.

Kemudian dalam tutorial ini, Anda akan menggunakan titik akses ini sebagai titik akses pendukung untuk Object Lambda Access Point Anda.

Untuk membuat titik akses

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pada panel navigasi di kiri, pilih Titik Akses.
3. Pada halaman Titik Akses, pilih Buat titik akses.
4. Di bidang Nama titik akses, masukkan nama (misalnya,**`tutorial-access-point`**) untuk titik akses.

   Untuk informasi lebih lanjut tentang penamaan titik akses, lihat Aturan penamaan titik akses Amazon S3.
5. Di bidang Nama Bucket, masukkan nama bucket yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**). S3 melampirkan titik akses ke bucket ini.

   (Opsional) Anda dapat memilih Browse S3 untuk menelusuri dan mencari bucket di akun Anda. Jika Anda memilih Browse S3, pilih bucket yang diinginkan, lalu pilih Pilih jalur untuk mengisi bidang nama Bucket dengan nama bucket tersebut.
6. Untuk asal Jaringan, pilih Internet.

   Untuk informasi lebih lanjut tentang asal jaringan untuk titik akses, lihat Membuat titik akses terbatas pada cloud privat virtual.
7. Secara default, semua pengaturan Blokir Akses Publik diaktifkan untuk titik akses Anda. Kami menyarankan agar Anda tetap mengaktifkan Blokir semua akses publik.

   Untuk informasi selengkapnya, lihat Mengelola akses publik ke titik akses.
8. Untuk semua pengaturan titik akses lainnya, pertahankan pengaturan default.

   (Opsional) Anda dapat mengubah pengaturan titik akses untuk mendukung kasus penggunaan Anda. Untuk tutorial ini, kami sarankan untuk menjaga pengaturan default.

(Opsional) Jika Anda perlu mengelola akses ke titik akses Anda, Anda dapat menentukan kebijakan titik akses. Untuk informasi selengkapnya, lihat Contoh kebijakan titik akses.

9. Pilih Buat titik akses.

Langkah 4: Buat fungsi Lambda

Untuk mengubah data asli, buat fungsi Lambda untuk digunakan dengan Titik Akses Lambda Objek S3 Anda.

Sublangkah

- Tulis kode fungsi Lambda dan buat paket penyebaran dengan lingkungan virtual
- Buat fungsi Lambda dengan peran eksekusi (konsol)
- Terapkan kode fungsi Lambda Anda dengan arsip file.zip dan konfigurasikan fungsi Lambda (konsol)

Tulis kode fungsi Lambda dan buat paket penyebaran dengan lingkungan virtual

1. Pada mesin lokal Anda, buat folder dengan nama folder `object-lambda` untuk lingkungan virtual untuk digunakan nanti dalam tutorial ini.
2. Di `object-lambda` folder, buat file dengan fungsi Lambda yang mengubah semua teks di objek asli menjadi huruf besar. Misalnya, Anda dapat menggunakan fungsi berikut yang ditulis dengan Python. Simpan fungsi ini dalam file bernama`transform.py`.

```
import boto3
import requests
from botocore.config import Config

# This function capitalizes all text in the original object
def lambda_handler(event, context):
    object_context = event["getObjectContext"]
    # Get the presigned URL to fetch the requested original object
    # from S3
    s3_url = object_context["inputS3Url"]
    # Extract the route and request token from the input context
    request_route = object_context["outputRoute"]
    request_token = object_context["outputToken"]
```

```
    # Get the original S3 object using the presigned URL
    response = requests.get(s3_url)
    original_object = response.content.decode("utf-8")

    # Transform all text in the original object to uppercase
    # You can replace it with your custom code based on your use case
    transformed_object = original_object.upper()

    # Write object back to S3 Object Lambda
    s3 = boto3.client('s3', config=Config(signature_version='s3v4'))
    # The WriteGetObjectResponse API sends the transformed data
    # back to S3 Object Lambda and then to the user
    s3.write_get_object_response(
        Body=transformed_object,
        RequestRoute=request_route,
        RequestToken=request_token)

    # Exit the Lambda function: return the status code
    return {'status_code': 200}
```

> **Note**
>
> Contoh sebelumnya fungsi Lambda memuat seluruh objek yang diminta ke dalam memori sebelum mengubahnya dan mengembalikannya ke klien. Atau, Anda dapat melakukan streaming objek dari S3 untuk menghindari memuat seluruh objek ke dalam memori. Pendekatan ini dapat berguna saat bekerja dengan benda besar. Untuk informasi selengkapnya tentang respons streaming dengan Titik Akses Objek Lambda, lihat contoh streaming di. Bekerja dengan `GetObject` permintaan di Lambda

Saat Anda menulis fungsi Lambda untuk digunakan dengan Titik Akses Lambda Objek S3, fungsi ini didasarkan pada konteks peristiwa masukan yang disediakan S3 Object Lambda ke fungsi Lambda. Konteks acara memberikan informasi tentang permintaan yang dibuat dalam acara yang diteruskan dari S3 Object Lambda ke Lambda. Ini berisi parameter yang Anda gunakan untuk membuat fungsi Lambda.

Bidang yang digunakan untuk membuat fungsi Lambda sebelumnya adalah sebagai berikut:

Bidang `getObjectContext` berarti rincian input dan output untuk koneksi ke Amazon S3 dan S3 Object Lambda. Ini memiliki bidang-bidang berikut:

- `inputS3Url`— URL presigned yang dapat digunakan fungsi Lambda untuk mengunduh objek asli dari titik akses pendukung. Dengan menggunakan URL yang telah ditetapkan sebelumnya, fungsi Lambda tidak perlu memiliki izin baca Amazon S3 untuk mengambil objek asli dan hanya dapat mengakses objek yang diproses oleh setiap pemanggilan.
- `outputRoute`— Token routing yang ditambahkan ke URL Lambda Objek S3 saat fungsi Lambda memanggil `WriteGetObjectResponse` untuk mengirim kembali objek yang diubah.
- `outputToken`— Token yang digunakan oleh S3 Object Lambda untuk `WriteGetObjectResponse` mencocokkan panggilan dengan penelepon asli saat mengirim kembali objek yang diubah.

Untuk informasi selengkapnya tentang semua bidang dalam konteks acara, lihat Format konteks peristiwa dan penggunaan danMenulis fungsi Lambda untuk Titik Akses S3 Lambda Objek.

3. Di terminal lokal Anda, masukkan perintah berikut untuk menginstal `virtualenv` paket:

```
python -m pip install virtualenv
```

4. Di terminal lokal Anda, buka `object-lambda` folder yang Anda buat sebelumnya, lalu masukkan perintah berikut untuk membuat dan menginisialisasi lingkungan virtual yang disebut`venv`.

```
python -m virtualenv venv
```

5. Untuk mengaktifkan lingkungan virtual, masukkan perintah berikut untuk menjalankan `activate` file dari folder lingkungan:

   Untuk pengguna macOS, jalankan perintah ini:

```
source venv/bin/activate
```

   Untuk pengguna Windows, jalankan perintah ini:

```
.\venv\Scripts\activate
```

   Sekarang, prompt perintah Anda berubah menjadi show (venv), menunjukkan bahwa lingkungan virtual aktif.

6. Untuk menginstal pustaka yang diperlukan, jalankan perintah berikut baris demi baris di lingkungan `venv` virtual.

   Perintah ini menginstal versi terbaru dari dependensi fungsi Lambda `lambda_handler` Anda. Dependensi ini adalah AWS SDK untuk Python (Boto3) dan modul request.

```
pip3 install boto3
```

```
pip3 install requests
```

7. Untuk menonaktifkan lingkungan virtual, jalankan perintah berikut:

```
deactivate
```

8. Untuk membuat paket penyebaran dengan pustaka yang diinstal sebagai `.zip` file bernama `lambda.zip` di root `object-lambda` direktori, jalankan perintah berikut baris demi baris di terminal lokal Anda.

   > **Tip**
   >
   > Perintah berikut mungkin perlu disesuaikan untuk bekerja di lingkungan khusus Anda. Misalnya, pustaka mungkin muncul di `site-packages` ataudist-packages, dan folder pertama mungkin `lib` ataulib64. Juga, `python` folder mungkin diberi nama dengan versi Python yang berbeda. Untuk menemukan paket tertentu, gunakan `pip show` perintah.

   Untuk pengguna macOS, jalankan perintah ini:

```
cd venv/lib/python3.8/site-packages
```

```
zip -r ../../../../lambda.zip .
```

   Untuk pengguna Windows, jalankan perintah ini:

```
cd .\venv\Lib\site-packages\
```

```
powershell Compress-Archive * ../../../lambda.zip
```

Perintah terakhir menyimpan paket deployment ke akar direktori `object-lambda`.

9. Tambahkan file kode fungsi `transform.py` ke root paket penyebaran Anda.

   Untuk pengguna macOS, jalankan perintah ini:

```
cd ../../../../
```

```
zip -g lambda.zip transform.py
```

   Untuk pengguna Windows, jalankan perintah ini:

```
cd ..\..\..\
```

```
powershell Compress-Archive -update transform.py lambda.zip
```

   Setelah menyelesaikan langkah ini, struktur direktori Anda adalah sebagai berikut:

```
lambda.zip$
  # transform.py
  # __pycache__
  | boto3/
  # certifi/
  # pip/
  # requests/
  ...
```

Buat fungsi Lambda dengan peran eksekusi (konsol)

1. Masuk ke AWS Management Console dan buka AWS Lambda konsol di https://console.aws.amazon.com/lambda/.

2. Di panel navigasi kiri, pilih Fungsi.

3. Pilih Buat fungsi.

4. Pilih Penulis dari scratch.
5. Di bagian Informasi dasar, lakukan hal berikut:
   a. Untuk Nama fungsi, masukkan **`tutorial-object-lambda-function`**.
   b. Untuk Runtime, pilih Python 3.8 atau versi yang lebih baru.
6. Perluas bagian Ubah peran eksekusi default. Di bawah Peran eksekusi, pilih Buat peran baru dengan izin Lambda dasar.

   Pada Langkah 5 nanti dalam tutorial ini, Anda melampirkan AmazonS3 ke peran eksekusi ObjectLambdaExecutionRolePolicy fungsi Lambda ini.
7. Pertahankan pengaturan yang tersisa disetel ke default.
8. Pilih Buat fungsi.

Terapkan kode fungsi Lambda Anda dengan arsip file.zip dan konfigurasikan fungsi Lambda (konsol)

1. Di AWS Lambda konsol di https://console.aws.amazon.com/lambda/, pilih Fungsi di panel navigasi kiri.
2. Pilih fungsi Lambda yang Anda buat sebelumnya (misalnya,**`tutorial-object-lambda-function`**).
3. Pada halaman detail fungsi Lambda, pilih tab Kode. Di bagian Sumber Kode, pilih Unggah dari dan kemudian file.zip.
4. Pilih Unggah untuk memilih `.zip` file lokal Anda.
5. Pilih `lambda.zip` file yang Anda buat sebelumnya, lalu pilih Buka.
6. Pilih Simpan.
7. Di bagian Pengaturan waktu proses, pilih Edit.
8. Pada halaman Edit pengaturan runtime, konfirmasikan bahwa Runtime diatur ke Python 3.8 atau versi yang lebih baru.
9. Untuk memberi tahu runtime Lambda metode handler mana dalam kode fungsi Lambda Anda yang akan dipanggil, masukkan untuk Handler. **`transform.lambda_handler`**

   Ketika Anda mengkonfigurasi fungsi dalam Python, nilai pengaturan handler adalah nama file dan nama modul handler, dipisahkan oleh titik. Misalnya, `transform.lambda_handler` memanggil `lambda_handler` metode yang didefinisikan dalam `transform.py` file.

10. Pilih Simpan.
11. (Opsional) Pada halaman detail fungsi Lambda Anda, pilih tab Konfigurasi. Di panel navigasi kiri, pilih Konfigurasi umum, lalu pilih Edit. Di bidang Timeout, masukkan **1** min **0** sec. Pertahankan pengaturan yang tersisa disetel ke default, dan pilih Simpan.

    Timeout adalah jumlah waktu yang memungkinkan Lambda menjalankan fungsi untuk pemanggilan sebelum menghentikannya. Default-nya adalah 3 detik. Durasi maksimum untuk fungsi Lambda yang digunakan oleh S3 Object Lambda adalah 60 detik. Harga didasarkan pada jumlah memori yang dikonfigurasi dan jumlah waktu kode Anda berjalan.

Langkah 5: Konfigurasikan kebijakan IAM untuk peran eksekusi fungsi Lambda Anda

Untuk mengaktifkan fungsi Lambda Anda untuk menyediakan data yang disesuaikan dan header respons kepada `GetObject` pemanggil, peran eksekusi fungsi Lambda Anda harus memiliki izin IAM untuk memanggil API. `WriteGetObjectResponse`

Untuk melampirkan kebijakan IAM ke peran fungsi Lambda Anda

1. Di AWS Lambda konsol di https://console.aws.amazon.com/lambda/, pilih Fungsi di panel navigasi kiri.
2. Pilih fungsi yang Anda buat di Langkah 4 (misalnya,**`tutorial-object-lambda-function`**).
3. Pada halaman detail fungsi Lambda Anda, pilih tab Konfigurasi, lalu pilih Izin di panel navigasi kiri.
4. Di bawah Peran eksekusi, pilih tautan nama Peran. Konsol IAM terbuka.
5. Pada halaman Ringkasan konsol IAM untuk peran eksekusi fungsi Lambda Anda, pilih tab Izin. Kemudian, dari menu Tambahkan Izin, pilih Lampirkan kebijakan.
6. Pada halaman Lampirkan Izin, masukkan **`AmazonS3ObjectLambdaExecutionRolePolicy`** di kotak pencarian untuk memfilter daftar kebijakan. Pilih kotak centang di sebelah nama kebijakan AmazonS3 ObjectLambdaExecutionRolePolicy.
7. Pilih Lampirkan kebijakan.

Langkah 6: Buat Titik Akses Lambda Objek S3

Titik Akses Lambda Objek S3 memberikan fleksibilitas untuk memanggil fungsi Lambda langsung dari permintaan S3 GET sehingga fungsi tersebut dapat memproses data yang diambil dari titik akses S3. Saat membuat dan mengonfigurasi Titik Akses Lambda Objek S3, Anda harus menentukan fungsi Lambda untuk memanggil dan memberikan konteks acara dalam format JSON sebagai parameter khusus untuk digunakan Lambda.

Untuk membuat Objek S3 Lambda Access Point

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilih Titik Akses Lambda Objek.
3. Pada halaman Titik Akses Lambda Objek, pilih Buat Titik Akses Lambda Objek.
4. Untuk nama Object Lambda Access Point, masukkan nama yang ingin Anda gunakan untuk Object Lambda Access Point (misalnya,). **`tutorial-object-lambda-accesspoint`**
5. Untuk Mendukung Titik Akses, masukkan atau telusuri ke titik akses standar yang Anda buat di Langkah 3 (misalnya,**`tutorial-access-point`**), lalu pilih Pilih Titik Akses pendukung.
6. Untuk API S3, untuk mengambil objek dari bucket S3 agar fungsi Lambda dapat diproses, pilih. GetObject
7. Untuk fungsi Invoke Lambda, Anda dapat memilih salah satu dari dua opsi berikut untuk tutorial ini.
   - Pilih Pilih dari fungsi di akun Anda, lalu pilih fungsi Lambda yang Anda buat di Langkah 4 (misalnya,**`tutorial-object-lambda-function`**) dari daftar dropdown fungsi Lambda.
   - Pilih Masukkan ARN, lalu masukkan Nama Sumber Daya Amazon (ARN) dari fungsi Lambda yang Anda buat di Langkah 4.
8. Untuk versi fungsi Lambda, pilih $LATEST (versi terbaru dari fungsi Lambda yang Anda buat di Langkah 4).
9. (Opsional) Jika Anda memerlukan fungsi Lambda Anda untuk mengenali dan memproses permintaan GET dengan header rentang dan nomor bagian, pilih Fungsi Lambda mendukung permintaan menggunakan rentang dan fungsi Lambda mendukung permintaan menggunakan nomor bagian. Jika tidak, kosongkan dua kotak centang ini.

   Untuk informasi selengkapnya tentang cara menggunakan nomor rentang atau bagian dengan S3 Object Lambda, lihat. Bekerja dengan header Range dan partNumber

10. (Opsional) Di bawah Payload - opsional, tambahkan teks JSON untuk menyediakan fungsi Lambda Anda dengan informasi tambahan.

    Payload adalah teks JSON opsional yang dapat Anda berikan ke fungsi Lambda Anda sebagai masukan untuk semua pemanggilan yang berasal dari Titik Akses Lambda Objek S3 tertentu. Untuk menyesuaikan perilaku untuk beberapa Titik Akses Lambda Objek yang menjalankan fungsi Lambda yang sama, Anda dapat mengonfigurasi muatan dengan parameter yang berbeda, sehingga memperluas fleksibilitas fungsi Lambda Anda.

    Untuk informasi selengkapnya tentang payload, lihatFormat konteks peristiwa dan penggunaan.
11. (Opsional) Untuk metrik Permintaan - opsional, pilih Nonaktifkan atau Aktifkan untuk menambahkan pemantauan Amazon S3 ke Titik Akses Lambda Objek Anda. Metrik permintaan ditagih dengan tarif Amazon CloudWatch standar. Untuk informasi selengkapnya, lihat harga CloudWatch.
12. Di bawah kebijakan Titik Akses Objek Lambda - opsional, pertahankan pengaturan default.

    (Opsional) Anda dapat menetapkan kebijakan sumber daya. Kebijakan resource ini memberikan izin `GetObject` API untuk menggunakan Object Lambda Access Point yang ditentukan.
13. Pertahankan pengaturan yang tersisa disetel ke default, dan pilih Buat Objek Lambda Access Point.

Langkah 7: Lihat data yang diubah

Sekarang, S3 Object Lambda siap untuk mengubah data Anda untuk kasus penggunaan Anda. Dalam tutorial ini, S3 Object Lambda mengubah semua teks dalam objek Anda menjadi huruf besar.

Sublangkah

- Lihat data yang diubah di Titik Akses Lambda Objek S3
- Jalankan skrip Python untuk mencetak data asli dan diubah

Lihat data yang diubah di Titik Akses Lambda Objek S3

Saat Anda meminta untuk mengambil file melalui Titik Akses Lambda Objek S3, Anda membuat panggilan `GetObject` API ke S3 Object Lambda. S3 Object Lambda memanggil fungsi Lambda untuk mengubah data Anda, dan kemudian mengembalikan data yang diubah sebagai respons terhadap panggilan API S3 standar. `GetObject`

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilih Titik Akses Lambda Objek.
3. Pada halaman Titik Akses Objek Lambda, pilih Titik Akses Lambda Objek S3 yang Anda buat di Langkah 6 (misalnya,). **`tutorial-object-lambda-accesspoint`**
4. Pada tab Objek pada Titik Akses Lambda Objek S3 Anda, pilih file yang memiliki nama yang sama (misalnya`tutorial.txt`,) dengan file yang Anda unggah ke bucket S3 di Langkah 2.

   File ini harus berisi semua data yang diubah.
5. Untuk melihat data yang diubah, pilih Buka atau Unduh.

Jalankan skrip Python untuk mencetak data asli dan diubah

Anda dapat menggunakan S3 Object Lambda dengan aplikasi yang ada. Untuk melakukannya, perbarui konfigurasi aplikasi Anda untuk menggunakan ARN Titik Akses Lambda Objek S3 baru yang Anda buat di Langkah 6 untuk mengambil data dari S3.

Contoh skrip Python berikut mencetak data asli dari bucket S3 dan data yang diubah dari S3 Object Lambda Access Point.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilih Titik Akses Lambda Objek.
3. Pada halaman Titik Akses Objek Lambda, pilih tombol radio di sebelah kiri Titik Akses Lambda Objek S3 yang Anda buat di Langkah 6 (misalnya,). **`tutorial-object-lambda-accesspoint`**
4. Pilih Salin ARN.
5. Simpan ARN untuk digunakan nanti.
6. Tulis skrip Python di mesin lokal Anda untuk mencetak data asli (misalnya,`tutorial.txt`) dari Bucket S3 Anda dan data yang diubah (misalnya,`tutorial.txt`) dari Titik Akses Lambda Objek S3 Anda). Anda dapat menggunakan contoh script berikut.

```
import boto3
from botocore.config import Config
```

```
s3 = boto3.client('s3', config=Config(signature_version='s3v4'))

def getObject(bucket, key):
    objectBody = s3.get_object(Bucket = bucket, Key = key)
    print(objectBody["Body"].read().decode("utf-8"))
    print("\n")

print('Original object from the S3 bucket:')
# Replace the two input parameters of getObject() below with
# the S3 bucket name that you created in Step 1 and
# the name of the file that you uploaded to the S3 bucket in Step 2
getObject("tutorial-bucket",
          "tutorial.txt")

print('Object transformed by S3 Object Lambda:')
# Replace the two input parameters of getObject() below with
# the ARN of your S3 Object Lambda Access Point that you saved earlier and
# the name of the file with the transformed data (which in this case is
# the same as the name of the file that you uploaded to the S3 bucket
# in Step 2)
getObject("arn:aws:s3-object-lambda:us-west-2:111122223333:accesspoint/tutorial-
object-lambda-accesspoint",
          "tutorial.txt")
```

7. Simpan skrip Python Anda dengan nama khusus (misalnya,`tutorial_print.py`) di folder (misalnya,`object-lambda`) yang Anda buat di Langkah 4 di mesin lokal Anda.
8. Di terminal lokal Anda, jalankan perintah berikut dari root direktori (misalnya,`object-lambda`) yang Anda buat di Langkah 4.

```
python3 tutorial_print.py
```

Anda akan melihat data asli dan data yang diubah (semua teks sebagai huruf besar) melalui terminal. Misalnya, Anda akan melihat sesuatu seperti teks berikut.

```
Original object from the S3 bucket:
Amazon S3 Object Lambda Tutorial:
You can add your own code to process data retrieved from S3 before
returning it to an application.

Object transformed by S3 Object Lambda:
AMAZON S3 OBJECT LAMBDA TUTORIAL:
YOU CAN ADD YOUR OWN CODE TO PROCESS DATA RETRIEVED FROM S3 BEFORE
```

```
RETURNING IT TO AN APPLICATION.
```

Langkah 8: Membersihkan

Jika Anda mengubah data Anda melalui S3 Object Lambda hanya sebagai latihan pembelajaran, hapus AWS sumber daya yang Anda alokasikan sehingga Anda tidak lagi menambah biaya.

Sublangkah

- Hapus Titik Akses Objek Lambda
- Hapus titik akses S3
- Hapus peran eksekusi untuk fungsi Lambda Anda
- Hapus fungsi Lambda
- Hapus grup CloudWatch log
- Hapus file asli di bucket sumber S3
- Hapus bucket sumber S3
- Hapus pengguna IAM

Hapus Titik Akses Objek Lambda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilih Titik Akses Lambda Objek.
3. Pada halaman Titik Akses Objek Lambda, pilih tombol radio di sebelah kiri Titik Akses Lambda Objek S3 yang Anda buat di Langkah 6 (misalnya,). **`tutorial-object-lambda-accesspoint`**
4. Pilih Hapus.
5. Konfirmasikan bahwa Anda ingin menghapus Titik Akses Objek Lambda Anda dengan memasukkan namanya di bidang teks yang muncul, lalu pilih Hapus.

Hapus titik akses S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Pada panel navigasi di kiri, pilih Titik Akses.
3. Arahkan ke titik akses yang Anda buat di Langkah 3 (misalnya,**`tutorial-access-point`**), dan pilih tombol radio di sebelah nama titik akses.
4. Pilih Hapus.
5. Konfirmasikan bahwa Anda ingin menghapus jalur akses Anda dengan memasukkan namanya di bidang teks yang muncul, lalu pilih Hapus.

Hapus peran eksekusi untuk fungsi Lambda Anda

1. Masuk ke AWS Management Console dan buka AWS Lambda konsol di https://console.aws.amazon.com/lambda/.
2. Di panel navigasi kiri, pilih Fungsi.
3. Pilih fungsi yang Anda buat di Langkah 4 (misalnya,**`tutorial-object-lambda-function`**).
4. Pada halaman detail fungsi Lambda Anda, pilih tab Konfigurasi, lalu pilih Izin di panel navigasi kiri.
5. Di bawah Peran eksekusi, pilih tautan nama Peran. Konsol IAM terbuka.
6. Pada halaman Ringkasan konsol IAM tentang peran eksekusi fungsi Lambda Anda, pilih Hapus peran.
7. Di kotak dialog Hapus peran, pilih Ya, hapus.

Hapus fungsi Lambda

1. Di AWS Lambda konsol di https://console.aws.amazon.com/lambda/, pilih Fungsi di panel navigasi kiri.
2. Pilih kotak centang di sebelah kiri nama fungsi yang Anda buat di Langkah 4 (misalnya,**`tutorial-object-lambda-function`**).
3. Pilih Tindakan, lalu pilih Hapus.
4. Di kotak dialog Hapus fungsi, pilih Hapus.

Hapus grup CloudWatch log

1. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Pada panel navigasi kiri, pilih Grup log.

3. Temukan grup log yang namanya diakhiri dengan fungsi Lambda yang Anda buat di Langkah 4 (misalnya,**`tutorial-object-lambda-function`**).
4. Pilih kotak centang di sebelah kiri nama grup log.
5. Pilih Tindakan, lalu pilih Hapus grup log.
6. Di kotak dialog Hapus grup log, pilih Hapus.

Hapus file asli di bucket sumber S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dalam daftar nama Bucket, pilih nama bucket tempat Anda mengunggah file asli di Langkah 2 (misalnya,**`tutorial-bucket`**).
4. Pilih kotak centang di sebelah kiri nama objek yang ingin Anda hapus (misalnya,`tutorial.txt`).
5. Pilih Hapus.
6. Pada halaman Hapus objek, di objek Hapus secara permanen? bagian, konfirmasikan bahwa Anda ingin menghapus objek ini dengan memasukkan **`permanently delete`** di kotak teks.
7. Pilih Hapus objek.

Hapus bucket sumber S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dalam daftar Bucket, pilih tombol radio di sebelah nama bucket yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**).
4. Pilih Hapus.
5. Di halaman Hapus bucket, konfirmasikan bahwa Anda ingin menghapus bucket dengan memasukkan nama bucket ke dalam bidang teks, lalu pilih Hapus bucket.

Hapus pengguna IAM

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi kiri, pilih Pengguna, lalu pilih kotak centang di sebelah nama pengguna yang ingin Anda hapus.
3. Pilih Hapus di bagian atas halaman.
4. Di Hapus ***nama pengguna***? kotak dialog, masukkan nama pengguna di bidang input teks untuk mengonfirmasi penghapusan pengguna. Pilih Hapus.

Langkah selanjutnya

Setelah menyelesaikan tutorial ini, Anda dapat menyesuaikan fungsi Lambda untuk kasus penggunaan Anda untuk memodifikasi data yang dikembalikan oleh permintaan GET S3 standar.

Berikut ini adalah daftar kasus penggunaan umum untuk S3 Object Lambda:

- Menyamarkan data sensitif untuk keamanan dan kepatuhan.

  Untuk informasi selengkapnya, lihat Tutorial: Mendeteksi dan menyunting data PII dengan S3 Object Lambda dan Amazon Comprehend.
- Memfilter baris data tertentu untuk memberikan informasi spesifik.
- Menambah data dengan informasi dari layanan atau database lain.
- Mengonversi seluruh format data, seperti mengonversi XHTML ke JSON untuk kompatibilitas aplikasi.
- Mengompresi atau mendekompresi file saat sedang diunduh.
- Mengubah ukuran dan menandai gambar.

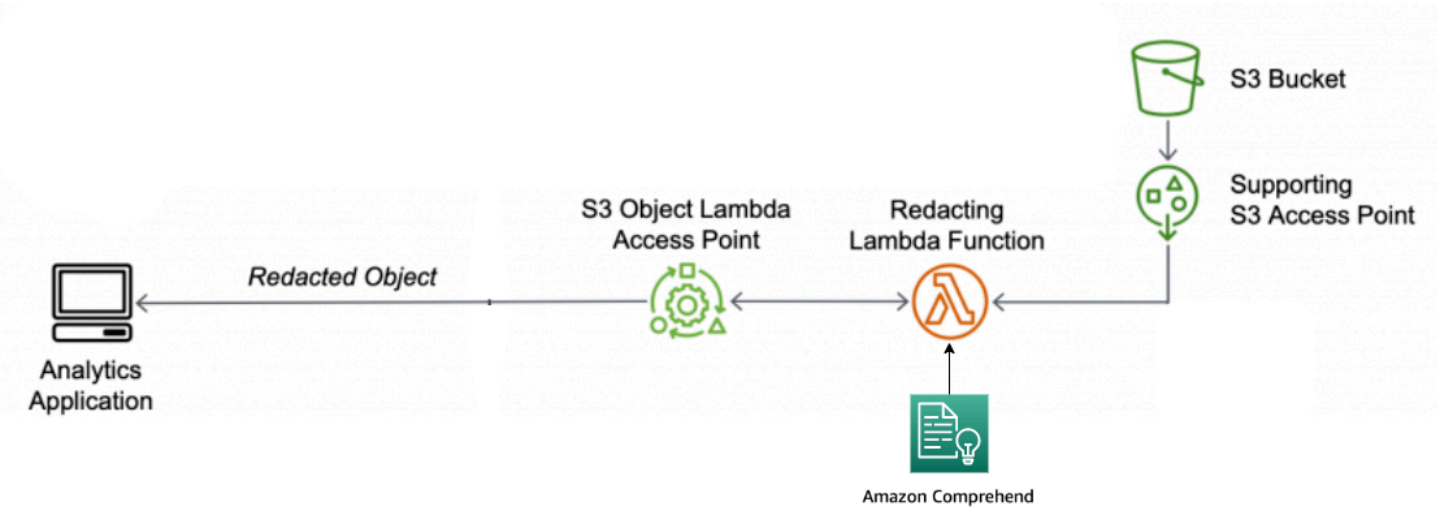
  Untuk informasi lebih lanjut, lihat Tutorial: Menggunakan S3 Object Lambda untuk menandai gambar secara dinamis saat diambil.
- Menerapkan aturan otorisasi khusus untuk mengakses data.

Untuk informasi selengkapnya tentang S3 Lambda Objek, lihat Mengubah objek dengan S3 Lambda Objek.

Tutorial: Mendeteksi dan menyunting data PII dengan S3 Object Lambda dan Amazon Comprehend

Saat Anda menggunakan Amazon S3 untuk kumpulan data bersama untuk beberapa aplikasi dan pengguna untuk mengakses, penting untuk membatasi informasi istimewa, seperti informasi identitas pribadi (PII), hanya untuk entitas yang berwenang. Misalnya, ketika aplikasi pemasaran menggunakan beberapa data yang mengandung PII, mungkin perlu terlebih dahulu menutupi data PII untuk memenuhi persyaratan privasi data. Juga, ketika aplikasi analitik menggunakan kumpulan data inventaris pesanan produksi, mungkin perlu terlebih dahulu menyunting informasi kartu kredit pelanggan untuk mencegah kebocoran data yang tidak diinginkan.

Dengan S3 Object Lambda dan fungsi AWS Lambda bawaan yang didukung oleh Amazon Comprehend, Anda dapat melindungi data PII yang diambil dari S3 sebelum mengembalikannya ke aplikasi. Secara khusus, Anda dapat menggunakan fungsi Lambda bawaan sebagai fungsi penyuntingan dan melampirkannya ke Titik Akses Lambda Objek S3. Ketika aplikasi (misalnya, aplikasi analitik) mengirimkan permintaan GET S3 standar, permintaan ini dibuat melalui Titik Akses Lambda Objek S3 memanggil fungsi Lambda penyuntingan bawaan untuk mendeteksi dan menyunting data PII yang diambil dari bucket S3 melalui jalur akses S3 yang mendukung. Kemudian, Titik Akses Lambda Objek S3 mengembalikan hasil yang disunting kembali ke aplikasi.



Dalam prosesnya, fungsi Lambda bawaan menggunakan Amazon Comprehend, layanan pemrosesan bahasa alami (NLP), untuk menangkap variasi dalam bagaimana PII direpresentasikan, terlepas dari bagaimana PII ada dalam teks (seperti numerik atau sebagai kombinasi kata dan angka). Amazon Comprehend bahkan dapat menggunakan konteks dalam teks untuk mengetahui apakah angka 4 digit adalah PIN, empat angka terakhir dari nomor Jaminan Sosial (SSN), atau satu tahun. Amazon Comprehend memproses file teks apa pun dalam format UTF-8 dan dapat melindungi

PII dalam skala besar tanpa memengaruhi akurasi. Untuk informasi selengkapnya, lihat Apa itu Amazon Comprehend? di Panduan Pengembang Amazon Comprehend.

Tujuan

Dalam tutorial ini, Anda belajar cara menggunakan S3 Object Lambda dengan fungsi Lambda prebuilt. `ComprehendPiiRedactionS3ObjectLambda` Fungsi ini menggunakan Amazon Comprehend untuk mendeteksi entitas PII. Kemudian menyunting entitas ini dengan menggantinya dengan tanda bintang. Dengan menyunting PII, Anda menyembunyikan data sensitif, yang dapat membantu keamanan dan kepatuhan.

Anda juga mempelajari cara menggunakan dan mengonfigurasi AWS Lambda fungsi bawaan AWS Serverless Application Repositoryuntuk bekerja sama dengan S3 Object Lambda untuk penyebaran yang mudah.

Topik

- Prasyarat: Buat pengguna IAM dengan izin
- Langkah 1: Buat ember S3
- Langkah 2: Unggah file ke bucket S3
- Langkah 3: Buat titik akses S3
- Langkah 4: Konfigurasikan dan terapkan fungsi Lambda bawaan
- Langkah 5: Buat Titik Akses Lambda Objek S3
- Langkah 6: Gunakan S3 Object Lambda Access Point untuk mengambil file yang disunting
- Langkah 7: Bersihkan
- Langkah selanjutnya

Prasyarat: Buat pengguna IAM dengan izin

Sebelum memulai tutorial ini, Anda harus memiliki AWS akun yang dapat Anda masuki sebagai AWS Identity and Access Management pengguna (pengguna IAM) dengan izin yang benar.

Anda dapat membuat pengguna IAM untuk tutorial. Untuk menyelesaikan tutorial ini, pengguna IAM Anda harus melampirkan kebijakan IAM berikut untuk mengakses AWS sumber daya yang relevan dan melakukan tindakan tertentu.

> **Note**
>
> Untuk kesederhanaan, tutorial ini membuat dan menggunakan pengguna IAM. Setelah menyelesaikan tutorial ini, ingatlah untukHapus pengguna IAM. Untuk penggunaan produksi, kami menyarankan Anda mengikuti praktik terbaik Keamanan di IAM dalam Panduan Pengguna IAM. Praktik terbaik mengharuskan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS dengan kredensi sementara. Praktik terbaik lainnya adalah meminta beban kerja untuk menggunakan kredensil sementara dengan peran IAM untuk mengakses. AWS Untuk mempelajari cara menggunakan AWS IAM Identity Center untuk membuat pengguna dengan kredensyal sementara, lihat Memulai di AWS IAM Identity Center Panduan Pengguna.
> Tutorial ini juga menggunakan kebijakan akses penuh. Untuk penggunaan produksi, sebaiknya Anda hanya memberikan izin minimum yang diperlukan untuk kasus penggunaan Anda, sesuai dengan praktik terbaik keamanan.

Pengguna IAM Anda memerlukan kebijakan AWS terkelola berikut:

- AmazonS3 FullAccess - Memberikan izin untuk semua tindakan Amazon S3, termasuk izin untuk membuat dan menggunakan Titik Akses Lambda Objek.
- AWSLambda_FullAccess— Memberikan izin untuk semua tindakan Lambda.
- AWSCloudFormationFullAccess— Memberikan izin untuk semua AWS CloudFormation tindakan.
- IAM FullAccess — Memberikan izin untuk semua tindakan IAM.
- IAM AccessAnalyzerReadOnlyAccess — Memberikan izin untuk membaca semua informasi akses yang disediakan oleh IAM Access Analyzer.

Anda dapat langsung melampirkan kebijakan yang ada saat membuat pengguna IAM. Untuk informasi selengkapnya tentang cara membuat pengguna IAM, lihat Membuat pengguna IAM (konsol) di Panduan Pengguna IAM.

Selain itu, pengguna IAM Anda memerlukan kebijakan yang dikelola pelanggan. Untuk memberikan izin pengguna IAM ke semua AWS Serverless Application Repository sumber daya dan tindakan, Anda harus membuat kebijakan IAM dan melampirkan kebijakan tersebut ke pengguna IAM.

Untuk membuat dan melampirkan kebijakan IAM ke pengguna IAM Anda

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di sebelah kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Pada tab Editor visual, untuk Layanan, pilih Pilih layanan. Kemudian, pilih Serverless Application Repository.
5. Untuk Tindakan, di bawah tindakan Manual, pilih Semua tindakan Repositori Aplikasi Tanpa Server (serverlessrepo: *) untuk tutorial ini.

   Sebagai praktik keamanan terbaik, Anda harus mengizinkan izin hanya untuk tindakan dan sumber daya yang dibutuhkan pengguna, berdasarkan kasus penggunaan Anda. Untuk informasi selengkapnya, lihat Praktik terbaik keamanan di IAM di Panduan Pengguna IAM.
6. Untuk Sumber Daya, pilih Semua sumber daya untuk tutorial ini.

   Sebagai praktik terbaik, Anda harus menentukan izin hanya untuk sumber daya tertentu di akun tertentu. Atau, Anda dapat memberikan hak istimewa paling sedikit menggunakan kunci kondisi. Untuk informasi selengkapnya, lihat Berikan hak istimewa paling sedikit di Panduan Pengguna IAM.
7. Pilih Berikutnya: Tanda.
8. Pilih Berikutnya: Tinjau.
9. Pada halaman Kebijakan ulasan, masukkan Nama (misalnya,**`tutorial-serverless-application-repository`**) dan Deskripsi (opsional) untuk kebijakan yang Anda buat. Tinjau ringkasan kebijakan untuk memastikan bahwa Anda telah memberikan izin yang dimaksud, lalu pilih Buat kebijakan untuk menyimpan kebijakan baru Anda.
10. Pada panel navigasi kiri, pilih Pengguna. Kemudian, pilih pengguna IAM untuk tutorial ini.
11. Pada halaman Ringkasan pengguna yang dipilih, pilih tab Izin, lalu pilih Tambahkan izin.
12. Di bawah Izin hibah, pilih Lampirkan kebijakan yang ada secara langsung.
13. Pilih kotak centang di samping kebijakan yang baru saja Anda buat (misalnya,**`tutorial-serverless-application-repository`**) lalu pilih Berikutnya: Tinjau.
14. Di bawah ringkasan Izin, tinjau ringkasan untuk memastikan bahwa Anda melampirkan kebijakan yang dimaksud. Kemudian, pilih Tambahkan izin.

Langkah 1: Buat ember S3

Buat bucket untuk menyimpan data asli yang Anda rencanakan untuk diubah.

Untuk membuat bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih Buat bucket.

   Halaman Buat bucket terbuka.
4. Untuk nama Bucket, masukkan nama (misalnya,**`tutorial-bucket`**) untuk bucket Anda.

   Untuk informasi selengkapnya tentang penamaan bucket di Amazon S3, lihat. Peraturan penamaan bucket
5. Untuk Wilayah, pilih Wilayah AWS tempat Anda ingin bucket berada.

   Untuk informasi selengkapnya tentang Wilayah bucket, lihatGambaran umum bucket.
6. Untuk pengaturan Blokir Akses Publik untuk bucket ini, pertahankan pengaturan default (Blokir semua akses publik diaktifkan).

   Kami menyarankan agar Anda tetap mengaktifkan semua pengaturan Blokir Akses Publik kecuali Anda perlu menonaktifkan satu atau beberapa pengaturan tersebut untuk kasus penggunaan Anda. Untuk informasi lebih lanjut tentang pemblokiran akses publik, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.
7. Untuk pengaturan yang tersisa, pertahankan defaultnya.

   (Opsional) Jika Anda ingin mengonfigurasi pengaturan bucket tambahan untuk kasus penggunaan spesifik Anda, lihatMembuat bucket.
8. Pilih Buat bucket.

Langkah 2: Unggah file ke bucket S3

Unggah file teks yang berisi data PII yang dikenal dari berbagai jenis, seperti nama, informasi perbankan, nomor telepon, dan SSN, ke bucket S3 sebagai data asli yang akan Anda edit PII nanti dalam tutorial ini.

Misalnya, Anda dapat mengunggah `tutorial.txt` file berikut. Ini adalah contoh file input dari Amazon Comprehend.

```
Hello Zhang Wei, I am John. Your AnyCompany Financial Services,
LLC credit card account 1111-0000-1111-0008 has a minimum payment
of $24.53 that is due by July 31st. Based on your autopay settings,
we will withdraw your payment on the due date from your
bank account number XXXXXX1111 with the routing number XXXXX0000.

Your latest statement was mailed to 100 Main Street, Any City,
WA 98121.
After your payment is received, you will receive a confirmation
text message at 206-555-0100.
If you have questions about your bill, AnyCompany Customer Service
is available by phone at 206-555-0199 or
email at support@anycompany.com.
```

Untuk mengunggah file ke ember

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dalam daftar Bucket, pilih nama bucket yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**) untuk mengunggah file Anda.
4. Di tab Objek untuk bucket Anda, pilih Unggah.
5. Pada halaman Unggah, di bawah File dan folder, pilih Tambahkan file.
6. Pilih file yang akan diunggah, lalu pilih Buka. Misalnya, Anda dapat mengunggah contoh `tutorial.txt` file yang disebutkan sebelumnya.
7. Pilih Unggah.

Langkah 3: Buat titik akses S3

Untuk menggunakan Titik Akses Lambda Objek S3 untuk mengakses dan mengubah data asli, Anda harus membuat titik akses S3 dan mengaitkannya dengan bucket S3 yang Anda buat di Langkah 1. Titik akses harus Wilayah AWS sama dengan objek yang ingin Anda ubah.

Kemudian dalam tutorial ini, Anda akan menggunakan titik akses ini sebagai titik akses pendukung untuk Object Lambda Access Point Anda.

Untuk membuat titik akses

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pada panel navigasi di kiri, pilih Titik Akses.
3. Pada halaman Titik Akses, pilih Buat titik akses.
4. Di bidang Nama titik akses, masukkan nama (misalnya,**`tutorial-pii-access-point`**) untuk titik akses.

   Untuk informasi lebih lanjut tentang penamaan titik akses, lihat Aturan penamaan titik akses Amazon S3.
5. Di bidang Nama Bucket, masukkan nama bucket yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**). S3 melampirkan titik akses ke bucket ini.

   (Opsional) Anda dapat memilih Browse S3 untuk menelusuri dan mencari bucket di akun Anda. Jika Anda memilih Browse S3, pilih bucket yang diinginkan, lalu pilih Pilih jalur untuk mengisi bidang nama Bucket dengan nama bucket tersebut.
6. Untuk asal Jaringan, pilih Internet.

   Untuk informasi lebih lanjut tentang asal jaringan untuk titik akses, lihat Membuat titik akses terbatas pada cloud privat virtual.
7. Secara default, semua pengaturan akses publik blok diaktifkan untuk titik akses Anda. Kami menyarankan agar Anda tetap mengaktifkan Blokir semua akses publik. Untuk informasi selengkapnya, lihat Mengelola akses publik ke titik akses.
8. Untuk semua pengaturan titik akses lainnya, pertahankan pengaturan default.

   (Opsional) Anda dapat mengubah pengaturan titik akses untuk mendukung kasus penggunaan Anda. Untuk tutorial ini, kami sarankan untuk menjaga pengaturan default.

   (Opsional) Jika Anda perlu mengelola akses ke titik akses Anda, Anda dapat menentukan kebijakan titik akses. Untuk informasi selengkapnya, lihat Contoh kebijakan titik akses.
9. Pilih Buat titik akses.

Langkah 4: Konfigurasikan dan terapkan fungsi Lambda bawaan

Untuk menyunting data PII, konfigurasikan dan terapkan AWS Lambda fungsi bawaan `ComprehendPiiRedactionS3ObjectLambda` untuk digunakan dengan Titik Akses Lambda Objek S3 Anda.

Untuk mengkonfigurasi dan menyebarkan fungsi Lambda

1. Masuk ke AWS Management Console dan lihat `ComprehendPiiRedactionS3ObjectLambda`fungsi di AWS Serverless Application Repository.
2. Untuk pengaturan Aplikasi, di bawah Nama aplikasi, pertahankan nilai default (`ComprehendPiiRedactionS3ObjectLambda`) untuk tutorial ini.

   (Opsional) Anda dapat memasukkan nama yang ingin Anda berikan ke aplikasi ini. Anda mungkin ingin melakukan ini jika Anda berencana untuk mengonfigurasi beberapa fungsi Lambda untuk kebutuhan akses yang berbeda untuk kumpulan data bersama yang sama.
3. Untuk MaskCharacter, pertahankan nilai default (**`*`**). Karakter topeng menggantikan setiap karakter dalam entitas PII yang disunting.
4. Untuk MaskMode, pertahankan nilai default (MASK). MaskModeNilai menentukan apakah entitas PII disunting dengan `MASK` karakter atau nilai. `PII_ENTITY_TYPE`
5. Untuk menyunting tipe data yang ditentukan, untuk PiiEntityTypes, pertahankan nilai default ALL. PiiEntityTypesNilai menentukan jenis entitas PII yang akan dipertimbangkan untuk redaksi.

   Untuk informasi selengkapnya tentang daftar jenis entitas PII yang didukung, lihat Mendeteksi Informasi Identifikasi Pribadi (PII) di Panduan Pengembang Amazon Comprehend.
6. Pertahankan pengaturan yang tersisa disetel ke default.

   (Opsional) Jika Anda ingin mengonfigurasi pengaturan tambahan untuk kasus penggunaan spesifik Anda, lihat bagian File Readme di sisi kiri halaman.
7. Pilih kotak centang di sebelah Saya mengakui bahwa aplikasi ini membuat peran IAM khusus.
8. Pilih Deploy.
9. Pada halaman aplikasi baru, di bawah Resources, pilih ID Logis dari fungsi Lambda yang Anda gunakan untuk meninjau fungsi pada halaman fungsi Lambda.

Langkah 5: Buat Titik Akses Lambda Objek S3

Titik Akses Lambda Objek S3 memberikan fleksibilitas untuk menjalankan fungsi Lambda langsung dari permintaan S3 GET sehingga fungsi tersebut dapat menyunting data PII yang diambil dari titik akses S3. Saat membuat dan mengonfigurasi Titik Akses Lambda Objek S3, Anda harus menentukan fungsi Lambda penyuntingan untuk memanggil dan menyediakan konteks acara dalam format JSON sebagai parameter khusus untuk digunakan Lambda.

Konteks acara memberikan informasi tentang permintaan yang dibuat dalam acara yang diteruskan dari S3 Object Lambda ke Lambda. Untuk informasi selengkapnya tentang semua bidang dalam konteks acara, lihatFormat konteks peristiwa dan penggunaan.

Untuk membuat Objek S3 Lambda Access Point

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilih Titik Akses Lambda Objek.
3. Pada halaman Titik Akses Lambda Objek, pilih Buat Titik Akses Lambda Objek.
4. Untuk nama Object Lambda Access Point, masukkan nama yang ingin Anda gunakan untuk Object Lambda Access Point (misalnya,). **`tutorial-pii-object-lambda-accesspoint`**
5. Untuk Mendukung Titik Akses, masukkan atau telusuri ke titik akses standar yang Anda buat di Langkah 3 (misalnya,**`tutorial-pii-access-point`**), lalu pilih Pilih Titik Akses pendukung.
6. Untuk API S3, untuk mengambil objek dari bucket S3 agar fungsi Lambda dapat diproses, pilih. GetObject
7. Untuk fungsi Invoke Lambda, Anda dapat memilih salah satu dari dua opsi berikut untuk tutorial ini.
   - Pilih Pilih dari fungsi di akun Anda dan pilih fungsi Lambda yang Anda gunakan di Langkah 4 (misalnya,**`serverlessrepo-ComprehendPiiRedactionS3ObjectLambda`**) dari daftar dropdown fungsi Lambda.
   - Pilih Masukkan ARN, lalu masukkan Nama Sumber Daya Amazon (ARN) dari fungsi Lambda yang Anda buat di Langkah 4.
8. Untuk versi fungsi Lambda, pilih $LATEST (versi terbaru dari fungsi Lambda yang Anda gunakan di Langkah 4).
9. (Opsional) Jika Anda memerlukan fungsi Lambda Anda untuk mengenali dan memproses permintaan GET dengan header rentang dan nomor bagian, pilih Fungsi Lambda mendukung

permintaan menggunakan rentang dan fungsi Lambda mendukung permintaan menggunakan nomor bagian. Jika tidak, kosongkan dua kotak centang ini.

Untuk informasi selengkapnya tentang cara menggunakan nomor rentang atau bagian dengan S3 Object Lambda, lihat. Bekerja dengan header Range dan partNumber

10. (Opsional) Di bawah Payload - opsional, tambahkan teks JSON untuk menyediakan fungsi Lambda Anda dengan informasi tambahan.

    Payload adalah teks JSON opsional yang dapat Anda berikan ke fungsi Lambda Anda sebagai masukan untuk semua pemanggilan yang berasal dari Titik Akses Lambda Objek S3 tertentu. Untuk menyesuaikan perilaku untuk beberapa Titik Akses Lambda Objek yang menjalankan fungsi Lambda yang sama, Anda dapat mengonfigurasi muatan dengan parameter yang berbeda, sehingga memperluas fleksibilitas fungsi Lambda Anda.

    Untuk informasi selengkapnya tentang payload, lihatFormat konteks peristiwa dan penggunaan.

11. (Opsional) Untuk metrik Permintaan - opsional, pilih Nonaktifkan atau Aktifkan untuk menambahkan pemantauan Amazon S3 ke Titik Akses Lambda Objek Anda. Metrik permintaan ditagih dengan tarif Amazon CloudWatch standar. Untuk informasi selengkapnya, lihat harga CloudWatch.

12. Di bawah kebijakan Titik Akses Objek Lambda - opsional, pertahankan pengaturan default.

    (Opsional) Anda dapat menetapkan kebijakan sumber daya. Kebijakan resource ini memberikan izin `GetObject` API untuk menggunakan Object Lambda Access Point yang ditentukan.

13. Pertahankan pengaturan yang tersisa disetel ke default, dan pilih Buat Objek Lambda Access Point.

Langkah 6: Gunakan S3 Object Lambda Access Point untuk mengambil file yang disunting

Sekarang, S3 Object Lambda siap untuk menyunting data PII dari file asli Anda.

Untuk menggunakan S3 Object Lambda Access Point untuk mengambil file yang disunting

Saat Anda meminta untuk mengambil file melalui Titik Akses Lambda Objek S3, Anda membuat panggilan `GetObject` API ke S3 Object Lambda. S3 Object Lambda memanggil fungsi Lambda untuk menyunting data PII Anda dan mengembalikan data yang diubah sebagai respons terhadap panggilan API S3 standar. `GetObject`

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilih Titik Akses Lambda Objek.
3. Pada halaman Titik Akses Objek Lambda, pilih Titik Akses Lambda Objek S3 yang Anda buat di Langkah 5 (misalnya,). **`tutorial-pii-object-lambda-accesspoint`**
4. Pada tab Objek pada Titik Akses Lambda Objek S3 Anda, pilih file yang memiliki nama yang sama (misalnya`tutorial.txt`,) dengan file yang Anda unggah ke bucket S3 di Langkah 2.

   File ini harus berisi semua data yang diubah.
5. Untuk melihat data yang diubah, pilih Buka atau Unduh.

   Anda harus dapat melihat file yang disunting, seperti yang ditunjukkan pada contoh berikut.

```
Hello *********. Your AnyCompany Financial Services,
LLC credit card account ******************* has a minimum payment
of $24.53 that is due by *********. Based on your autopay settings,
we will withdraw your payment on the due date from your
bank account ********** with the routing number *********.

Your latest statement was mailed to **********************************.
After your payment is received, you will receive a confirmation
text message at ************.
If you have questions about your bill, AnyCompany Customer Service
is available by phone at ************ or
email at **********************.
```

Langkah 7: Bersihkan

Jika Anda menyunting data Anda melalui S3 Object Lambda hanya sebagai latihan pembelajaran, hapus AWS sumber daya yang Anda alokasikan sehingga Anda tidak lagi dikenakan biaya.

Sublangkah

- Hapus Titik Akses Objek Lambda
- Hapus titik akses S3
- Hapus fungsi Lambda
- Hapus grup CloudWatch log

- Hapus file asli di bucket sumber S3
- Hapus bucket sumber S3
- Hapus peran IAM untuk fungsi Lambda Anda
- Menghapus kebijakan terkelola pelanggan untuk pengguna IAM Anda
- Hapus pengguna IAM

Hapus Titik Akses Objek Lambda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilih Titik Akses Lambda Objek.
3. Pada halaman Titik Akses Objek Lambda, pilih tombol opsi di sebelah kiri Titik Akses Lambda Objek S3 yang Anda buat di Langkah 5 (misalnya,). **`tutorial-pii-object-lambda-accesspoint`**
4. Pilih Hapus.
5. Konfirmasikan bahwa Anda ingin menghapus Titik Akses Objek Lambda Anda dengan memasukkan namanya di bidang teks yang muncul, lalu pilih Hapus.

Hapus titik akses S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pada panel navigasi di kiri, pilih Titik Akses.
3. Arahkan ke titik akses yang Anda buat di Langkah 3 (misalnya,**`tutorial-pii-access-point`**), dan pilih tombol opsi di sebelah nama titik akses.
4. Pilih Hapus.
5. Konfirmasikan bahwa Anda ingin menghapus jalur akses Anda dengan memasukkan namanya di bidang teks yang muncul, lalu pilih Hapus.

Hapus fungsi Lambda

1. Di AWS Lambda konsol di https://console.aws.amazon.com/lambda/, pilih Fungsi di panel navigasi kiri.

2. Pilih fungsi yang Anda buat di Langkah 4 (misalnya,**`serverlessrepo-ComprehendPiiRedactionS3ObjectLambda`**).
3. Pilih Tindakan, lalu pilih Hapus.
4. Di kotak dialog Hapus fungsi, pilih Hapus.

Hapus grup CloudWatch log

1. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Pada panel navigasi kiri, pilih Grup log.
3. Temukan grup log yang namanya diakhiri dengan fungsi Lambda yang Anda buat di Langkah 4 (misalnya,**`serverlessrepo-ComprehendPiiRedactionS3ObjectLambda`**).
4. Pilih Tindakan, lalu pilih Hapus grup log.
5. Di kotak dialog Hapus grup log, pilih Hapus.

Hapus file asli di bucket sumber S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dalam daftar nama Bucket, pilih nama bucket tempat Anda mengunggah file asli di Langkah 2 (misalnya,**`tutorial-bucket`**).
4. Pilih kotak centang di sebelah kiri nama objek yang ingin Anda hapus (misalnya,`tutorial.txt`).
5. Pilih Hapus.
6. Pada halaman Hapus objek, di objek Hapus secara permanen? bagian, konfirmasikan bahwa Anda ingin menghapus objek ini dengan memasukkan **`permanently delete`** di kotak teks.
7. Pilih Hapus objek.

Hapus bucket sumber S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.

3. Dalam daftar Bucket, pilih tombol opsi di sebelah nama bucket yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**).
4. Pilih Hapus.
5. Di halaman Hapus bucket, konfirmasikan bahwa Anda ingin menghapus bucket dengan memasukkan nama bucket ke dalam bidang teks, lalu pilih Hapus bucket.

Hapus peran IAM untuk fungsi Lambda Anda

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi kiri, pilih Peran, lalu pilih kotak centang di samping nama peran yang ingin Anda hapus. Nama peran dimulai dengan nama fungsi Lambda yang Anda gunakan di Langkah 4 (misalnya,). **`serverlessrepo-ComprehendPiiRedactionS3ObjectLambda`**
3. Pilih Hapus.
4. Di kotak dialog Hapus, masukkan nama peran di bidang input teks untuk mengonfirmasi penghapusan. Lalu, pilih Hapus.

Menghapus kebijakan terkelola pelanggan untuk pengguna IAM Anda

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di sebelah kiri, pilih Kebijakan.
3. Pada halaman Kebijakan, masukkan nama kebijakan terkelola pelanggan yang Anda buat di Prasyarat (misalnya,**`tutorial-serverless-application-repository`**) di kotak pencarian untuk memfilter daftar kebijakan. Pilih tombol opsi di sebelah nama kebijakan yang ingin Anda hapus.
4. Pilih Tindakan, lalu pilih Hapus.
5. Konfirmasikan bahwa Anda ingin menghapus kebijakan ini dengan memasukkan namanya di bidang teks yang muncul, lalu pilih Hapus.

Hapus pengguna IAM

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.

2. Di panel navigasi kiri, pilih Pengguna, lalu pilih kotak centang di sebelah nama pengguna yang ingin Anda hapus.
3. Pilih Hapus di bagian atas halaman.
4. Di Hapus ***nama pengguna***? kotak dialog, masukkan nama pengguna di bidang input teks untuk mengonfirmasi penghapusan pengguna. Pilih Hapus.

Langkah selanjutnya

Setelah menyelesaikan tutorial ini, Anda dapat menjelajahi lebih lanjut kasus penggunaan terkait berikut:

- Anda dapat membuat beberapa Titik Akses Lambda Objek S3 dan mengaktifkannya dengan fungsi Lambda bawaan yang dikonfigurasi secara berbeda untuk menyunting jenis PII tertentu tergantung pada kebutuhan bisnis aksesori data.

  Setiap jenis pengguna mengasumsikan peran IAM dan hanya memiliki akses ke satu Titik Akses Lambda Objek S3 (dikelola melalui kebijakan IAM). Kemudian, Anda melampirkan setiap fungsi `ComprehendPiiRedactionS3ObjectLambda` Lambda yang dikonfigurasi untuk kasus penggunaan redaksi yang berbeda ke Titik Akses Lambda Objek S3 yang berbeda. Untuk setiap Titik Akses Lambda Objek S3, Anda dapat memiliki jalur akses S3 pendukung untuk membaca data dari bucket S3 yang menyimpan kumpulan data bersama.

  Untuk informasi selengkapnya tentang cara membuat kebijakan bucket S3 yang memungkinkan pengguna membaca dari bucket hanya melalui jalur akses S3, lihat. Mengonfigurasi kebijakan IAM untuk menggunakan titik akses

  Untuk informasi selengkapnya tentang cara memberikan izin pengguna untuk mengakses fungsi Lambda, titik akses S3, dan Titik Akses Lambda Objek S3, lihat. Mengonfigurasi kebijakan IAM untuk Titik Akses Lambda Objek

- Anda dapat membangun fungsi Lambda Anda sendiri dan menggunakan S3 Object Lambda dengan fungsi Lambda yang disesuaikan untuk memenuhi kebutuhan data spesifik Anda.

  Misalnya, untuk menjelajahi berbagai nilai data, Anda dapat menggunakan Lambda Objek S3 dan fungsi Lambda Anda sendiri yang menggunakan fitur Amazon Comprehend tambahan, seperti pengenalan entitas, pengenalan frasa kunci, analisis sentimen, dan klasifikasi dokumen, untuk memproses data. Anda juga dapat menggunakan S3 Object Lambda bersama dengan Amazon

Comprehend Medical, layanan NLP yang memenuhi syarat HIPAA, untuk menganalisis dan mengekstrak data dengan cara yang sadar konteks.

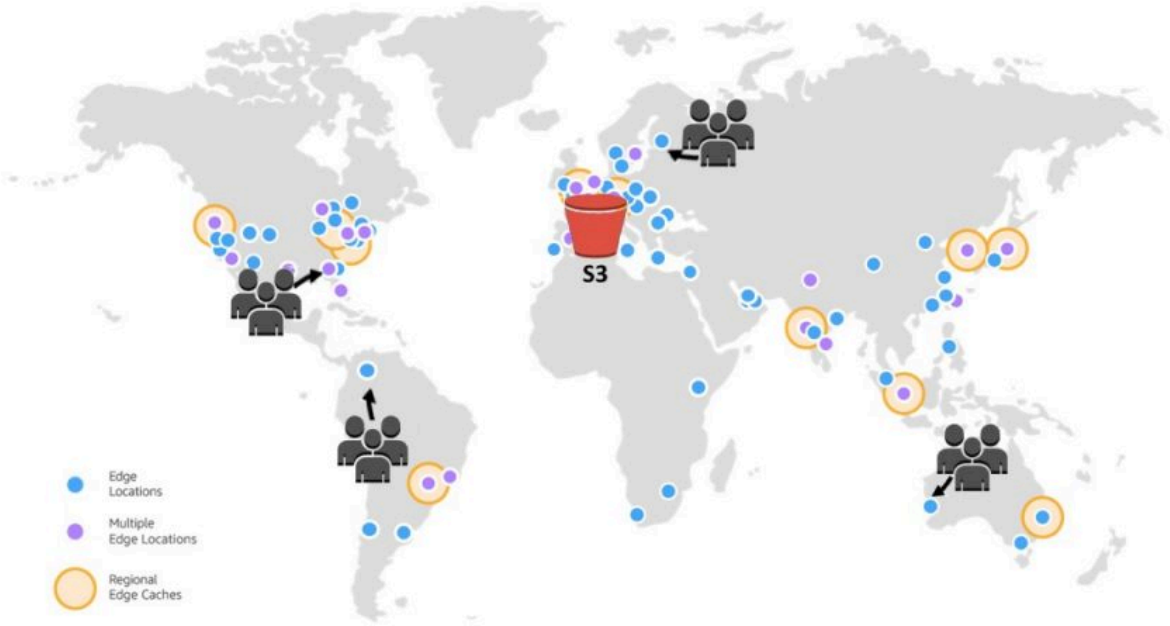
Untuk informasi selengkapnya tentang cara mengubah data dengan S3 Object Lambda dan fungsi Lambda Anda sendiri, lihat. Tutorial: Mengubah data untuk aplikasi Anda dengan S3 Object Lambda

Tutorial: Hosting video streaming sesuai permintaan dengan Amazon S3, Amazon, dan CloudFront Amazon Route 53

Anda dapat menggunakan Amazon S3 dengan Amazon CloudFront untuk meng-host video untuk dilihat sesuai permintaan dengan cara yang aman dan terukur. Streaming video on demand (VOD) berarti konten video Anda disimpan di server dan pemirsa dapat menontonnya kapan saja.

CloudFront adalah layanan jaringan pengiriman konten (CDN) yang cepat, sangat aman, dan dapat diprogram. CloudFront dapat mengirimkan konten Anda dengan aman melalui HTTPS dari semua lokasi CloudFront tepi di seluruh dunia. Untuk informasi selengkapnya CloudFront, lihat Apa itu Amazon CloudFront? di Panduan CloudFront Pengembang Amazon.

CloudFront caching mengurangi jumlah permintaan yang harus ditanggapi oleh server asal Anda secara langsung. Saat pemirsa (pengguna akhir) meminta video yang Anda tayang CloudFront, permintaan tersebut diarahkan ke lokasi tepi terdekat yang lebih dekat ke tempat penampil berada. CloudFront menyajikan video dari cache-nya, mengambilnya dari bucket S3 hanya jika belum di-cache. Fitur manajemen caching ini mempercepat pengiriman video Anda ke pemirsa secara global dengan latensi rendah, throughput tinggi, dan kecepatan transfer tinggi. Untuk informasi selengkapnya tentang manajemen CloudFront caching, lihat Mengoptimalkan caching dan ketersediaan di Panduan Pengembang Amazon CloudFront .



Tujuan

Dalam tutorial ini, Anda mengonfigurasi bucket S3 untuk meng-host streaming video sesuai permintaan menggunakan CloudFront untuk pengiriman dan Amazon Route 53 untuk Sistem Nama Domain (DNS) dan manajemen domain khusus.

Topik

- Prasyarat: Daftarkan dan konfigurasikan domain khusus dengan Route 53
- Langkah 1: Buat ember S3
- Langkah 2: Unggah video ke bucket S3
- Langkah 3: Buat identitas akses CloudFront asal
- Langkah 4: Buat CloudFront distribusi
- Langkah 5: Akses video melalui CloudFront distribusi
- Langkah 6: Konfigurasikan CloudFront distribusi Anda untuk menggunakan nama domain kustom Anda
- Langkah 7: Akses video S3 melalui CloudFront distribusi dengan nama domain khusus
- (Opsional) Langkah 8: Lihat data tentang permintaan yang diterima oleh CloudFront distribusi Anda

- Langkah 9: Membersihkan
- Langkah selanjutnya

Prasyarat: Daftarkan dan konfigurasikan domain khusus dengan Route 53

Sebelum Anda memulai tutorial ini, Anda harus mendaftar dan mengkonfigurasi domain kustom (misalnya,**`example.com`**) dengan Route 53 sehingga Anda dapat mengonfigurasi CloudFront distribusi Anda untuk menggunakan nama domain khusus nanti.

Tanpa nama domain khusus, video S3 Anda dapat diakses publik dan di-host melalui CloudFront URL yang terlihat mirip dengan berikut ini:

```
https://CloudFront distribution domain name/Path to an S3 video
```

Misalnya, **`https://d111111abcdef8.cloudfront.net/sample.mp4`**.

Setelah Anda mengonfigurasi CloudFront distribusi Anda untuk menggunakan nama domain khusus yang dikonfigurasi dengan Route 53, video S3 Anda dapat diakses publik dan di-host melalui CloudFront URL yang terlihat mirip dengan berikut ini:

```
https://CloudFront distribution alternate domain name/Path to an S3 video
```

Misalnya, **`https://www.example.com/sample.mp4`**. Nama domain khusus lebih sederhana dan lebih intuitif untuk digunakan pemirsa Anda.

Untuk mendaftarkan domain khusus, lihat Mendaftarkan domain baru menggunakan Route 53 di Panduan Pengembang Amazon Route 53.

Ketika Anda mendaftarkan nama domain dengan Route 53, Route 53 membuat zona host untuk Anda, yang akan Anda gunakan nanti dalam tutorial ini. Zona yang dihosting ini adalah tempat Anda menyimpan informasi tentang cara merutekan lalu lintas untuk domain Anda, misalnya, ke instans Amazon EC2 atau distribusi. CloudFront

Ada biaya yang terkait dengan pendaftaran domain, zona host Anda, dan kueri DNS yang diterima oleh domain Anda. Untuk informasi lebih lanjut, lihat Harga Amazon Route 53.

> **Note**
>
> Ketika Anda mendaftarkan domain, biayanya segera dan itu tidak dapat diubah. Anda dapat memilih untuk tidak memperbarui domain secara otomatis, tetapi Anda membayar di muka dan memilikinya untuk tahun ini. Untuk informasi selengkapnya, lihat Mendaftarkan nama domain di Panduan Pengembang Amazon Route 53.

Langkah 1: Buat ember S3

Buat ember untuk menyimpan video asli yang Anda rencanakan untuk streaming.

Untuk membuat bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih Buat bucket.

   Halaman Buat bucket terbuka.
4. Untuk nama Bucket, masukkan nama untuk bucket Anda (misalnya,**`tutorial-bucket`**).

   Untuk informasi selengkapnya tentang penamaan bucket di Amazon S3, lihat. Peraturan penamaan bucket
5. Untuk Wilayah, pilih Wilayah AWS tempat Anda ingin ember berada.

   Jika memungkinkan, Anda harus memilih Wilayah yang paling dekat dengan mayoritas pemirsa Anda. Untuk informasi selengkapnya tentang Wilayah bucket, lihatGambaran umum bucket.
6. Untuk pengaturan Blokir Akses Publik untuk bucket ini, pertahankan pengaturan default (Blokir semua akses publik diaktifkan).

   Bahkan dengan Blokir semua akses publik diaktifkan, pemirsa masih dapat mengakses video yang diunggah. CloudFront Fitur ini merupakan keuntungan utama menggunakan CloudFront untuk meng-host video yang disimpan di S3.

   Kami menyarankan agar Anda tetap mengaktifkan semua pengaturan kecuali Anda perlu mematikan satu atau beberapa pengaturan untuk kasus penggunaan Anda. Untuk informasi

lebih lanjut tentang pemblokiran akses publik, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

7. Untuk pengaturan yang tersisa, pertahankan defaultnya.

   (Opsional) Jika Anda ingin mengonfigurasi pengaturan bucket tambahan untuk kasus penggunaan spesifik Anda, lihatMembuat bucket.

8. Pilih Buat bucket.

Langkah 2: Unggah video ke bucket S3

Prosedur berikut menjelaskan cara mengunggah file video ke bucket S3 dengan menggunakan konsol. Jika Anda mengunggah banyak file video besar ke S3, Anda mungkin ingin menggunakan Amazon S3 Transfer Acceleration untuk mengonfigurasi transfer file yang cepat dan aman. Transfer Acceleration dapat mempercepat pengunggahan video ke bucket S3 Anda untuk transfer jarak jauh video yang lebih besar. Untuk informasi selengkapnya, lihat Mengonfigurasi transfer file yang cepat dan aman menggunakan Amazon S3 Transfer Acceleration.

Untuk mengunggah file ke bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dalam daftar Bucket, pilih nama bucket yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**) untuk mengunggah file Anda.
4. Di tab Objek untuk bucket Anda, pilih Unggah.
5. Pada halaman Unggah, di bawah File dan folder, pilih Tambahkan file.
6. Pilih file yang akan diunggah, lalu pilih Buka.

   Misalnya, Anda dapat mengunggah file video bernama`sample.mp4`.

7. Pilih Unggah.

Langkah 3: Buat identitas akses CloudFront asal

Untuk membatasi akses langsung ke video dari bucket S3 Anda, buat CloudFront pengguna khusus yang disebut identitas akses asal (OAI). Anda akan mengaitkan OAI dengan distribusi Anda nanti dalam tutorial ini. Dengan menggunakan OAI, Anda memastikan bahwa pemirsa tidak dapat mem-

bypass CloudFront dan mendapatkan video langsung dari bucket S3. Hanya CloudFront OAI yang dapat mengakses file di bucket S3. Untuk informasi selengkapnya, lihat Membatasi akses ke konten Amazon S3 dengan menggunakan OAI di Panduan Pengembang Amazon CloudFront .

Untuk membuat CloudFront OAI

1. Masuk ke AWS Management Console dan buka CloudFront konsol dihttps://console.aws.amazon.com/cloudfront/v4/home.
2. Di panel navigasi kiri, di bawah bagian Keamanan, pilih Akses asal.
3. Di bawah tab Identitas, pilih Buat identitas akses asal.
4. Masukkan nama (misalnya,**`S3-OAI`**) untuk identitas akses asal yang baru.
5. Pilih Buat.

Langkah 4: Buat CloudFront distribusi

Untuk digunakan CloudFront untuk melayani dan mendistribusikan video di bucket S3 Anda, Anda harus membuat CloudFront distribusi.

Sublangkah

- Buat CloudFront distribusi
- Tinjau kebijakan bucket

Buat CloudFront distribusi

1. Masuk ke AWS Management Console dan buka CloudFront konsol dihttps://console.aws.amazon.com/cloudfront/v4/home.
2. Di panel navigasi kiri, pilih Distribusi.
3. Pilih Buat Distribusi.
4. Di bagian Origin, untuk domain Origin, pilih nama domain asal S3 Anda, yang dimulai dengan nama bucket S3 yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**).
5. Untuk akses Origin, pilih Identitas akses lama.
6. Di bawah Origin Access Identity, pilih identitas akses asal yang Anda buat di Langkah 3 (misalnya,**`S3-OAI`**).
7. Di bawah kebijakan Bucket, pilih Ya, perbarui kebijakan bucket.

8. Di bagian perilaku cache default, di bawah kebijakan protokol Viewer, pilih Redirect HTTP ke HTTPS.

   Ketika Anda memilih fitur ini, permintaan HTTP secara otomatis dialihkan ke HTTPS untuk mengamankan situs web Anda dan melindungi data pemirsa Anda.
9. Untuk pengaturan lain di bagian perilaku cache default, pertahankan nilai default.

   (Opsional) Anda dapat mengontrol berapa lama file Anda tetap dalam CloudFront cache sebelum CloudFront meneruskan permintaan lain ke asal Anda. Mengurangi durasi memungkinkan Anda untuk melayani konten dinamis. Meningkatkan durasi berarti pemirsa Anda mendapatkan kinerja yang lebih baik karena file Anda lebih mungkin ditayangkan langsung dari cache tepi. Durasi yang lebih lama juga mengurangi beban yang berasal dari Anda. Untuk informasi selengkapnya, lihat Mengelola berapa lama konten tetap berada di cache (kedaluwarsa) di Panduan CloudFront Pengembang Amazon.
10. Untuk bagian lain, pertahankan pengaturan yang tersisa disetel ke default.

    Untuk informasi selengkapnya tentang opsi pengaturan yang berbeda, lihat Nilai yang Anda Tentukan Saat Membuat atau Memperbarui Distribusi di Panduan CloudFront Pengembang Amazon.
11. Di bagian bawah halaman, pilih Buat distribusi.
12. Pada tab Umum untuk CloudFront distribusi Anda, di bawah Detail, nilai kolom terakhir yang dimodifikasi untuk distribusi Anda berubah dari Deploying ke stempel waktu saat distribusi terakhir diubah. Proses ini biasanya memakan waktu beberapa menit.

Tinjau kebijakan bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dalam daftar Bucket, pilih nama bucket yang Anda gunakan sebelumnya sebagai asal CloudFront distribusi Anda (misalnya,**`tutorial-bucket`**).
4. Pilih tab Izin.
5. Di bagian Kebijakan Bucket, konfirmasikan bahwa Anda melihat pernyataan yang mirip dengan yang berikut ini dalam teks kebijakan bucket:

```
{
```

```
    "Version": "2008-10-17",
    "Id": "PolicyForCloudFrontPrivateContent",
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access
 Identity EH1HDMB1FH2TC"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::tutorial-bucket/*"
        }
    ]
}
```

Ini adalah pernyataan bahwa CloudFront distribusi Anda ditambahkan ke kebijakan bucket saat Anda memilih Ya, perbarui kebijakan bucket lebih awal.

Pembaruan kebijakan bucket ini menunjukkan bahwa Anda berhasil mengonfigurasi CloudFront distribusi untuk membatasi akses ke bucket S3. Karena pembatasan ini, objek dalam ember hanya dapat diakses melalui CloudFront distribusi Anda.

Langkah 5: Akses video melalui CloudFront distribusi

Sekarang, CloudFront dapat menyajikan video yang disimpan di bucket S3 Anda. Untuk mengakses video Anda CloudFront, Anda harus menggabungkan nama domain CloudFront distribusi Anda dengan jalur ke video di bucket S3.

Untuk membuat URL ke video S3 menggunakan nama domain CloudFront distribusi

1. Masuk ke AWS Management Console dan buka CloudFront konsol dihttps://console.aws.amazon.com/cloudfront/v4/home.
2. Di panel navigasi kiri, pilih Distribusi.
3. Untuk mendapatkan nama domain distribusi, lakukan hal berikut:
   a. Di kolom Origins, temukan CloudFront distribusi yang benar dengan mencari nama asalnya, yang dimulai dengan bucket S3 yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**).

   b. Setelah menemukan distribusi dalam daftar, perluas kolom Nama domain untuk menyalin nilai nama domain untuk CloudFront distribusi Anda.

4. Di tab browser baru, tempel nama domain distribusi yang Anda salin.
5. Kembali ke tab browser sebelumnya, dan buka konsol S3 di https://console.aws.amazon.com/s3/.
6. Di panel navigasi kiri, pilih Bucket.
7. Dalam daftar Bucket, pilih nama bucket yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**).
8. Dalam daftar Objek, pilih nama video yang Anda unggah di Langkah 2 (misalnya,`sample.mp4`).
9. Pada halaman detail objek, di bagian Ikhtisar objek, salin nilai Kunci. Nilai ini adalah jalur ke objek video yang diunggah di bucket S3.
10. Kembali ke tab browser tempat Anda sebelumnya menempelkan nama domain distribusi, masukkan garis miring ke depan (**/**) setelah nama domain distribusi, lalu tempel jalur ke video yang Anda salin sebelumnya (misalnya,). `sample.mp4`

    Sekarang, video S3 Anda dapat diakses publik dan di-host melalui CloudFront URL yang terlihat mirip dengan berikut ini:

    ```
    https://CloudFront distribution domain name/Path to the S3 video
    ```

    Ganti *`nama domain CloudFront distribusi`* dan *`Path ke video S3`* dengan nilai yang sesuai. Contoh URL adalah**`https://d111111abcdef8.cloudfront.net/sample.mp4`**.

Langkah 6: Konfigurasikan CloudFront distribusi Anda untuk menggunakan nama domain kustom Anda

Untuk menggunakan nama domain Anda sendiri alih-alih nama CloudFront domain di URL untuk mengakses video S3, tambahkan nama domain alternatif ke CloudFront distribusi Anda.

Sublangkah

- Minta sertifikat SSL
- Tambahkan nama domain alternatif ke CloudFront distribusi Anda
- Buat catatan DNS untuk merutekan lalu lintas dari nama domain alternatif Anda ke nama domain CloudFront distribusi Anda

- Periksa apakah IPv6 diaktifkan untuk distribusi Anda dan buat catatan DNS lain jika diperlukan

Minta sertifikat SSL

Untuk memungkinkan pemirsa menggunakan HTTPS dan nama domain khusus Anda di URL untuk streaming video Anda, gunakan AWS Certificate Manager (ACM) untuk meminta sertifikat Secure Sockets Layer (SSL). Sertifikat SSL menetapkan koneksi jaringan terenkripsi ke situs web.

1. Masuk ke AWS Management Console dan buka konsol ACM di https://console.aws.amazon.com/acm/.
2. Jika halaman pengantar muncul, di bawah Sertifikat penyediaan, pilih Memulai.
3. Pada halaman Minta sertifikat, pilih Minta sertifikat publik, lalu pilih Minta sertifikat.
4. Pada halaman Tambahkan nama domain, masukkan nama domain yang memenuhi syarat (FQDN) dari situs yang ingin Anda amankan dengan sertifikat SSL/TLS. Anda dapat menggunakan tanda bintang (`*`) untuk meminta sertifikat wildcard untuk melindungi beberapa nama situs dalam domain yang sama. Untuk tutorial ini, masukkan **`*`** dan nama domain kustom yang Anda konfigurasikan di Prasyarat. Misalnya, masukkan**`*.example.com`**, lalu pilih Berikutnya.

   Untuk informasi selengkapnya, lihat Untuk meminta sertifikat publik ACM (konsol) di Panduan AWS Certificate Manager Pengguna.
5. Pada halaman Pilih metode validasi, pilih validasi DNS. Lalu, pilih Selanjutnya.

   Jika Anda dapat mengedit konfigurasi DNS Anda, kami sarankan Anda menggunakan validasi domain DNS daripada validasi email. Validasi DNS memiliki banyak manfaat dibandingkan validasi email. Untuk informasi selengkapnya, lihat Opsi 1: Validasi DNS di AWS Certificate Manager Panduan Pengguna.
6. (Opsional) Pada halaman Tambahkan tag, beri tag sertifikat Anda dengan metadata.
7. Pilih Tinjau.
8. Pada halaman Tinjauan, verifikasi bahwa informasi di bawah nama Domain dan metode Validasi sudah benar. Kemudian, pilih Konfirmasi dan minta.

   Halaman Validasi menunjukkan bahwa permintaan Anda sedang diproses dan domain sertifikat sedang divalidasi. Sertifikat menunggu validasi berada dalam status validasi Tertunda.
9. Pada halaman Validasi, pilih panah bawah di sebelah kiri nama domain kustom Anda, lalu pilih Buat catatan di Route 53 untuk memvalidasi kepemilikan domain Anda melalui DNS.

Melakukan hal ini menambahkan catatan CNAME yang disediakan oleh AWS Certificate Manager ke konfigurasi DNS Anda.

10. Dalam Buat catatan di Route 53 kotak dialog, pilih Buat.

    Halaman Validasi harus menampilkan pemberitahuan status Sukses di bagian bawah.

11. Pilih Lanjutkan untuk melihat halaman daftar Sertifikat.

    Status untuk sertifikat baru Anda berubah dari validasi Tertunda menjadi Diterbitkan dalam waktu 30 menit.

Tambahkan nama domain alternatif ke CloudFront distribusi Anda

1. Masuk ke AWS Management Console dan buka CloudFront konsol dihttps://console.aws.amazon.com/cloudfront/v4/home.
2. Di panel navigasi kiri, pilih Distribusi.
3. Pilih ID untuk distribusi yang Anda buat di Langkah 4.
4. Pada tab Umum, buka bagian Pengaturan, dan pilih Edit.
5. Pada halaman Edit pengaturan, untuk Nama domain alternatif (CNAME) - opsional, pilih Tambahkan item untuk menambahkan nama domain khusus yang ingin Anda gunakan di URL untuk video S3 yang disajikan oleh distribusi ini CloudFront .

   Dalam tutorial ini, misalnya, jika Anda ingin merutekan lalu lintas untuk subdomain, sepertiwww`.example.com`, masukkan nama subdomain (`www`) dengan nama domain (`example.com`). Secara khusus, masukkan**`www.example.com`**.

   > **Note**
   >
   > Nama domain alternatif (CNAME) yang Anda tambahkan harus dicakup oleh sertifikat SSL yang sebelumnya Anda lampirkan ke distribusi Anda CloudFront.

6. Untuk sertifikat SSL Kustom - opsional, pilih sertifikat SSL yang Anda minta sebelumnya (misalnya,**`*.example.com`**).

> **Note**
>
> Jika Anda tidak melihat sertifikat SSL segera setelah Anda memintanya, tunggu 30 menit, lalu segarkan daftar hingga sertifikat SSL tersedia untuk Anda pilih.

7. Pertahankan pengaturan yang tersisa disetel ke default. Pilih Simpan perubahan.
8. Pada tab Umum untuk distribusi, tunggu nilai Terakhir diubah dari Deploying ke stempel waktu saat distribusi terakhir diubah.

Buat catatan DNS untuk merutekan lalu lintas dari nama domain alternatif Anda ke nama domain CloudFront distribusi Anda

1. Masuk ke AWS Management Console dan buka konsol Route 53 di https://console.aws.amazon.com/route53/.
2. Di panel navigasi kiri, pilih Zona yang dihosting.
3. Pada halaman Zona yang di-host, pilih nama zona yang dihosting yang dibuat Route 53 untuk Anda di Prasyarat (misalnya,). **`example.com`**
4. Pilih Buat catatan, lalu gunakan metode Quick create record.
5. Untuk nama Rekam, pertahankan nilai untuk nama rekaman sama dengan nama domain alternatif dari CloudFront distribusi yang Anda tambahkan sebelumnya.

   Dalam tutorial ini, untuk merutekan lalu lintas ke subdomain, sepertiwww`.example.com`, masukkan nama subdomain tanpa nama domain. Misalnya, masukkan hanya **`www`** di bidang teks sebelum nama domain kustom Anda.
6. Untuk jenis Rekam, pilih A - Rute lalu lintas ke alamat IPv4 dan beberapa AWS sumber daya.
7. Untuk Nilai, pilih sakelar Alias untuk mengaktifkan sumber daya alias.
8. Di bawah Rute lalu lintas ke, pilih Alias untuk CloudFront didistribusikan dari daftar dropdown.
9. Di kotak pencarian yang bertuliskan Pilih distribusi, pilih nama domain CloudFront distribusi yang Anda buat di Langkah 4.

   Untuk menemukan nama domain CloudFront distribusi Anda, lakukan hal berikut:

   a. Di tab browser baru, masuk ke AWS Management Console dan buka CloudFront konsol dihttps://console.aws.amazon.com/cloudfront/v3/home.
   b. Di panel navigasi kiri, pilih Distribusi.

c. Di kolom Origins, temukan CloudFront distribusi yang benar dengan mencari nama asalnya, yang dimulai dengan bucket S3 yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**).

d. Setelah menemukan distribusi dalam daftar, perluas kolom Nama domain untuk melihat nilai nama domain untuk CloudFront distribusi Anda.

10. Pada halaman Buat catatan di konsol Route 53, untuk pengaturan yang tersisa, pertahankan defaultnya.

11. Pilih Create records (Buat catatan).

Periksa apakah IPv6 diaktifkan untuk distribusi Anda dan buat catatan DNS lain jika diperlukan

Jika IPv6 diaktifkan untuk distribusi Anda, Anda harus membuat catatan DNS lain.

1. Untuk memeriksa apakah IPv6 diaktifkan untuk distribusi Anda, lakukan hal berikut:

   a. Masuk ke AWS Management Console dan buka CloudFront konsol dihttps://console.aws.amazon.com/cloudfront/v4/home.

   b. Di panel navigasi kiri, pilih Distribusi.

   c. Pilih ID CloudFront distribusi yang Anda buat di Langkah 4.

   d. Pada tab Umum, di bawah Pengaturan, periksa apakah IPv6 diatur ke Diaktifkan.

   Jika IPv6 diaktifkan untuk distribusi Anda, Anda harus membuat catatan DNS lain.

2. Jika IPv6 diaktifkan untuk distribusi Anda, lakukan hal berikut untuk membuat catatan DNS:

   a. Masuk ke AWS Management Console dan buka konsol Route 53 di https://console.aws.amazon.com/route53/.

   b. Di panel navigasi kiri, pilih Zona yang dihosting.

   c. Pada halaman Zona yang di-host, pilih nama zona yang dihosting yang dibuat Route 53 untuk Anda di Prasyarat (misalnya,). **`example.com`**

   d. Pilih Buat catatan, lalu gunakan metode Quick create record.

   e. Untuk nama Rekam, di bidang teks sebelum nama domain kustom Anda, ketikkan nilai yang sama dengan yang Anda ketik saat membuat catatan DNS IPv4 sebelumnya. Misalnya, dalam tutorial ini, untuk merutekan lalu lintas untuk subdomain`www.example.com`, masukkan saja**`www`**.

f. Untuk jenis Rekam, pilih AAAA - Rute lalu lintas ke alamat IPv6 dan beberapa sumber daya. AWS

g. Untuk Nilai, pilih sakelar Alias untuk mengaktifkan sumber daya alias.

h. Di bawah Rute lalu lintas ke, pilih Alias untuk CloudFront didistribusikan dari daftar dropdown.

i. Di kotak pencarian yang bertuliskan Pilih distribusi, pilih nama domain CloudFront distribusi yang Anda buat di Langkah 4.

j. Untuk pengaturan yang tersisa, pertahankan defaultnya.

k. Pilih Create records (Buat catatan).

Langkah 7: Akses video S3 melalui CloudFront distribusi dengan nama domain khusus

Untuk mengakses video S3 menggunakan URL khusus, Anda harus menggabungkan nama domain alternatif Anda dengan jalur ke video di bucket S3.

Untuk membuat URL khusus untuk mengakses video S3 melalui distribusi CloudFront

1. Masuk ke AWS Management Console dan buka CloudFront konsol dihttps://console.aws.amazon.com/cloudfront/v4/home.
2. Di panel navigasi kiri, pilih Distribusi.
3. Untuk mendapatkan nama domain alternatif dari CloudFront distribusi Anda, lakukan hal berikut:

   a. Di kolom Origins, temukan CloudFront distribusi yang benar dengan mencari nama asalnya, yang dimulai dengan nama bucket S3 untuk bucket yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**).

   b. Setelah menemukan distribusi dalam daftar, perluas kolom Nama domain alternatif untuk menyalin nilai nama domain alternatif CloudFront distribusi Anda.
4. Di tab browser baru, tempel nama domain alternatif CloudFront distribusi.
5. Kembali ke tab browser sebelumnya, dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
6. Temukan jalur ke video S3 Anda, seperti yang dijelaskan pada Langkah 5.
7. Kembali ke tab browser tempat Anda sebelumnya menempelkan nama domain alternatif, masukkan garis miring (**/**), lalu tempel jalur ke video S3 Anda (misalnya,). `sample.mp4`

Sekarang, video S3 Anda dapat diakses publik dan di-host melalui CloudFront URL khusus yang terlihat mirip dengan berikut ini:

```
https://CloudFront distribution alternate domain name/Path to the S3 video
```

Ganti *CloudFront distribusi nama domain alternatif* dan *Path ke video S3* dengan nilai yang sesuai. Contoh URL adalah**https://www.example.com/sample.mp4**.

(Opsional) Langkah 8: Lihat data tentang permintaan yang diterima oleh CloudFront distribusi Anda

Untuk melihat data tentang permintaan yang diterima oleh CloudFront distribusi Anda

1. Masuk ke AWS Management Console dan buka CloudFront konsol dihttps://console.aws.amazon.com/cloudfront/v4/home.
2. Di panel navigasi kiri, di bawah Laporan & analitik, pilih laporan dari konsol, mulai dari statistik Cache, Objek Populer, Perujuk Teratas, Penggunaan, dan Pemirsa.

   Anda dapat memfilter setiap dasbor laporan. Untuk informasi selengkapnya, lihat CloudFront Laporan di Konsol di Panduan CloudFront Pengembang Amazon.
3. Untuk memfilter data, pilih ID CloudFront distribusi yang Anda buat di Langkah 4.

Langkah 9: Membersihkan

Jika Anda meng-host video streaming S3 menggunakan CloudFront dan Route 53 hanya sebagai latihan pembelajaran, hapus AWS sumber daya yang Anda alokasikan sehingga Anda tidak lagi dikenakan biaya.

> **Note**
>
> Ketika Anda mendaftarkan domain, biayanya segera dan itu tidak dapat diubah. Anda dapat memilih untuk tidak memperbarui domain secara otomatis, tetapi Anda membayar di muka dan memilikinya untuk tahun ini. Untuk informasi selengkapnya, lihat Mendaftarkan nama domain di Panduan Pengembang Amazon Route 53.

Sublangkah

- Hapus CloudFront distribusi
- Hapus catatan DNS
- Hapus zona yang dihosting publik untuk domain kustom Anda
- Hapus nama domain kustom dari Route 53
- Hapus video asli di bucket sumber S3
- Hapus bucket sumber S3

Hapus CloudFront distribusi

1. Masuk ke AWS Management Console dan buka CloudFront konsol dihttps://console.aws.amazon.com/cloudfront/v4/home.
2. Di panel navigasi kiri, pilih Distribusi.
3. Di kolom Origins, temukan CloudFront distribusi yang benar dengan mencari nama asalnya, yang dimulai dengan nama bucket S3 untuk bucket yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**).
4. Untuk menghapus CloudFront distribusi, Anda harus menonaktifkannya terlebih dahulu.
   - Jika nilai kolom Status Diaktifkan dan nilai Terakhir diubah adalah stempel waktu saat distribusi terakhir diubah, lanjutkan untuk menonaktifkan distribusi sebelum menghapusnya.
   - Jika nilai Status Diaktifkan dan nilai Terakhir diubah adalah Deploying, tunggu hingga nilai Status berubah menjadi stempel waktu saat distribusi terakhir diubah. Kemudian lanjutkan untuk menonaktifkan distribusi sebelum menghapusnya.
5. Untuk menonaktifkan CloudFront distribusi, lakukan hal berikut:
   a. Dalam daftar Distribusi, pilih kotak centang di sebelah ID untuk distribusi yang ingin Anda hapus.
   b. Untuk menonaktifkan distribusi, pilih Nonaktifkan, lalu pilih Nonaktifkan untuk mengonfirmasi.

      Jika Anda menonaktifkan distribusi yang memiliki nama domain alternatif yang terkait dengannya, CloudFront berhenti menerima lalu lintas untuk nama domain tersebut (sepertiwww`.example.com`), bahkan jika distribusi lain memiliki nama domain alternatif dengan wildcard (`*`) yang cocok dengan domain yang sama (seperti`*.example.com`).

c. Nilai Status segera berubah menjadi Dinonaktifkan. Tunggu hingga nilai Perubahan terakhir diubah dari Deploying ke stempel waktu saat distribusi terakhir diubah.

   Karena CloudFront harus menyebarkan perubahan ini ke semua lokasi tepi, mungkin perlu beberapa menit sebelum pembaruan selesai dan opsi Hapus tersedia bagi Anda untuk menghapus distribusi.

6. Untuk menghapus distribusi yang dinonaktifkan, lakukan hal berikut:

   a. Pilih kotak centang di sebelah ID untuk distribusi yang ingin Anda hapus.
   b. Pilih Hapus, lalu pilih Hapus untuk mengonfirmasi.

Hapus catatan DNS

Jika Anda ingin menghapus zona yang dihosting publik untuk domain (termasuk catatan DNS), lihat Hapus zona yang dihosting publik untuk domain kustom Anda di Panduan Pengembang Amazon Route 53. Jika Anda hanya ingin menghapus catatan DNS yang dibuat pada Langkah 6, lakukan hal berikut:

1. Masuk ke AWS Management Console dan buka konsol Route 53 di https://console.aws.amazon.com/route53/.
2. Di panel navigasi kiri, pilih Zona yang dihosting.
3. Pada halaman Zona yang di-host, pilih nama zona yang dihosting yang dibuat Route 53 untuk Anda di Prasyarat (misalnya,). **`example.com`**
4. Dalam daftar catatan, pilih kotak centang di samping catatan yang ingin Anda hapus (catatan yang Anda buat di Langkah 6).

   > **Note**
   >
   > Anda tidak dapat menghapus catatan yang memiliki nilai Type NS atau SOA.

5. Pilih Hapus catatan.
6. Untuk mengonfirmasi penghapusan, pilih Hapus.

   Perubahan pada catatan membutuhkan waktu untuk menyebar ke server DNS Route 53. Saat ini, satu-satunya cara untuk memverifikasi bahwa perubahan Anda telah disebarkan adalah dengan menggunakan tindakan GetChange API. Perubahan biasanya menyebar ke semua server nama Route 53 dalam waktu 60 detik.

Hapus zona yang dihosting publik untuk domain kustom Anda

> **Warning**
>
> Jika Anda ingin menyimpan pendaftaran domain tetapi berhenti merutekan lalu lintas internet ke situs web atau aplikasi web Anda, kami sarankan Anda menghapus catatan di zona yang dihosting (seperti yang dijelaskan di bagian sebelumnya) alih-alih menghapus zona yang dihosting.
> Jika Anda menghapus zona yang dihosting, orang lain dapat menggunakan domain dan merutekan lalu lintas ke sumber daya mereka sendiri menggunakan nama domain Anda. Selain itu, jika Anda menghapus zona yang dihosting, Anda tidak dapat membatalkan menghapusnya. Anda harus membuat zona host baru dan memperbarui server nama untuk pendaftaran domain Anda, yang dapat memakan waktu hingga 48 jam untuk diterapkan.
> Jika Anda ingin membuat domain tidak tersedia di internet, pertama-tama Anda dapat mentransfer layanan DNS Anda ke layanan DNS gratis dan kemudian menghapus zona yang dihosting Route 53. Hal ini mencegah kueri DNS agar tidak salah dirutekan di masa mendatang.
>
> 1. Jika domain terdaftar di Route 53, lihat Menambahkan atau mengubah server nama dan merekatkan catatan untuk domain di Panduan Pengembang Amazon Route 53 untuk informasi tentang cara mengganti server nama Route 53 dengan server nama untuk layanan DNS baru.
> 2. Jika domain yang terdaftar dengan registrar lain, gunakan metode yang disediakan oleh registrar untuk mengubah server nama domain.
>
> > **Note**
> >
> > Jika Anda menghapus zona yang dihosting untuk subdomain (`www.example.com`), Anda tidak perlu mengubah server nama untuk domain ()`example.com`.

1. Masuk ke AWS Management Console dan buka konsol Route 53 di https://console.aws.amazon.com/route53/.
2. Di panel navigasi kiri, pilih Zona yang dihosting.
3. Pada halaman Zona yang dihosting, pilih nama zona yang dihosting yang ingin Anda hapus.

4. Pada tab Rekaman untuk zona yang dihosting, konfirmasikan bahwa zona yang dihosting yang ingin Anda hapus hanya berisi catatan NS dan SOA.

   Jika berisi catatan tambahan, hapus terlebih dahulu.

   Jika Anda membuat catatan NS untuk subdomain di zona yang dihosting, hapus catatan tersebut juga.
5. Pada tab penandatanganan DNSSEC untuk zona yang dihosting, nonaktifkan penandatanganan DNNSSEC jika diaktifkan. Untuk informasi selengkapnya, lihat Menonaktifkan penandatanganan DNSSEC di Panduan Pengembang Amazon Route 53.
6. Di bagian atas halaman detail zona yang dihosting, pilih Hapus zona.
7. Untuk mengonfirmasi penghapusan, masukkan**`delete`**, lalu pilih Hapus.

Hapus nama domain kustom dari Route 53

Untuk sebagian besar domain tingkat atas (TLD), Anda dapat menghapus pendaftaran jika tidak lagi menginginkannya. Jika Anda menghapus pendaftaran nama domain dari Route 53 sebelum pendaftaran dijadwalkan kedaluwarsa, AWS tidak mengembalikan biaya pendaftaran. Untuk informasi selengkapnya, lihat Menghapus pendaftaran nama domain di Panduan Pengembang Amazon Route 53.

> **Important**
>
> Jika Anda ingin mentransfer domain antara Akun AWS atau mentransfer domain ke registrar lain, jangan hapus domain dan berharap untuk segera mendaftarkannya kembali. Sebagai gantinya, lihat dokumentasi yang berlaku di Panduan Pengembang Amazon Route 53:
>
> - Mentransfer domain ke yang berbeda Akun AWS
> - Mentransfer domain dari Amazon Route 53 ke registrar lain

Hapus video asli di bucket sumber S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.

3. Dalam daftar nama Bucket, pilih nama bucket tempat Anda mengunggah video di Langkah 2 (misalnya,**`tutorial-bucket`**).
4. Pada tab Objek, pilih kotak centang di sebelah nama objek yang ingin Anda hapus (misalnya,`sample.mp4`).
5. Pilih Hapus.
6. Di bawah Hapus objek secara permanen? , masukkan **`permanently delete`** untuk mengonfirmasi bahwa Anda ingin menghapus objek ini.
7. Pilih Hapus objek.

Hapus bucket sumber S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dalam daftar Bucket, pilih tombol opsi di sebelah nama bucket yang Anda buat di Langkah 1 (misalnya,**`tutorial-bucket`**).
4. Pilih Hapus.
5. Di halaman Hapus bucket, konfirmasikan bahwa Anda ingin menghapus bucket dengan memasukkan nama bucket ke dalam bidang teks, lalu pilih Hapus bucket.

Langkah selanjutnya

Setelah Anda menyelesaikan tutorial ini, Anda dapat menjelajahi lebih lanjut kasus penggunaan terkait berikut:

- Transkode video S3 ke dalam format streaming yang dibutuhkan oleh televisi tertentu atau perangkat yang terhubung sebelum menghosting video ini dengan CloudFront distribusi.

  Untuk menggunakan Operasi Batch Amazon S3, AWS Lambda dan AWS Elemental MediaConvert mentranskode kumpulan video ke berbagai format media keluaran, lihat. Tutorial: Video transcoding batch dengan Operasi Batch S3,, dan AWS LambdaAWS Elemental MediaConvert
- Host objek lain yang disimpan di S3, seperti gambar, audio, grafik gerak, style sheet, HTML, JavaScript, aplikasi React, dan sebagainya, menggunakan CloudFront dan Route 53.

Misalnya, lihat Tutorial: Mengonfigurasi situs web statis menggunakan domain kustom yang terdaftar di Route 53 danMempercepat situs web Anda dengan Amazon CloudFront.

- Gunakan Amazon S3 Transfer Acceleration untuk mengonfigurasi transfer file yang cepat dan aman. Transfer Acceleration dapat mempercepat pengunggahan video ke bucket S3 Anda untuk transfer jarak jauh video yang lebih besar. Transfer Acceleration meningkatkan kinerja transfer dengan merutekan lalu lintas melalui lokasi tepi yang didistribusikan CloudFront secara global dan melalui AWS jaringan backbone. Ini juga menggunakan optimasi protokol jaringan. Untuk informasi selengkapnya, lihat Mengonfigurasi transfer file yang cepat dan aman menggunakan Amazon S3 Transfer Acceleration.

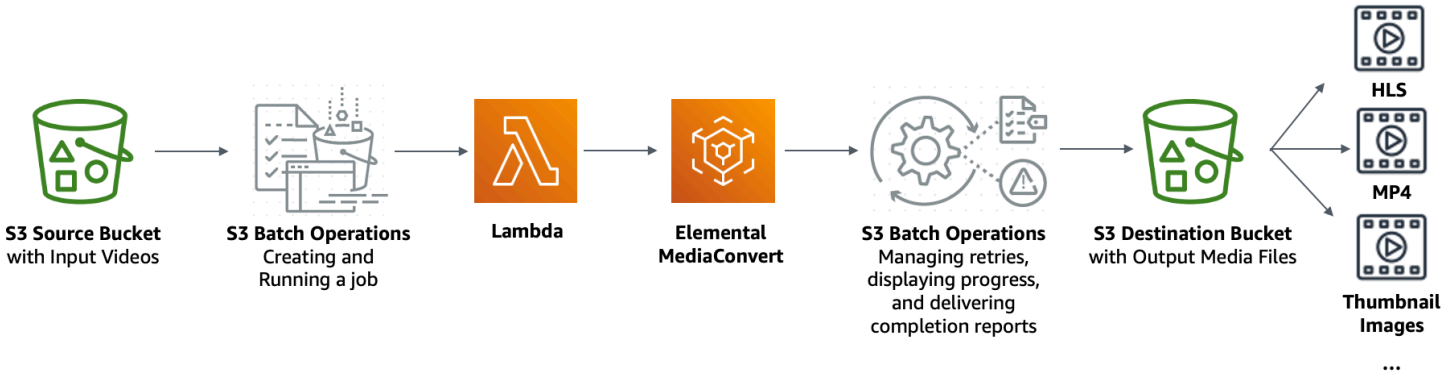
Tutorial: Video transcoding batch dengan Operasi Batch S3,, dan AWS LambdaAWS Elemental MediaConvert

Konsumen video menggunakan perangkat dari segala bentuk, ukuran, dan vintage untuk menikmati konten media. Beragam perangkat ini menghadirkan tantangan bagi pembuat konten dan distributor. Alih-alih dalam one-size-fits-all format, video harus dikonversi sehingga dapat menjangkau berbagai ukuran, format, dan bitrate. Tugas konversi ini bahkan lebih menantang ketika Anda memiliki sejumlah besar video yang harus dikonversi.

AWS menawarkan Anda metode untuk membangun arsitektur terdistribusi yang dapat diskalakan yang melakukan hal berikut:

- Menelan video masukan
- Memproses video untuk pemutaran di berbagai perangkat
- Menyimpan file media yang ditranskode
- Mengirimkan file media keluaran untuk memenuhi permintaan

Jika Anda memiliki repositori video ekstensif yang disimpan di Amazon S3, Anda dapat mentranskode video ini dari format sumbernya ke beberapa jenis file dalam ukuran, resolusi, dan format yang diperlukan oleh pemutar video atau perangkat tertentu. Secara khusus, Operasi Batch S3 memberi Anda solusi untuk menjalankan AWS Lambda fungsi untuk video input yang ada di bucket sumber S3. Kemudian, fungsi Lambda memanggil AWS Elemental MediaConvertuntuk melakukan tugas transcoding video skala besar. File media keluaran yang dikonversi disimpan dalam bucket tujuan S3.



Tujuan

Dalam tutorial ini, Anda mempelajari cara mengatur Operasi Batch S3 untuk menjalankan fungsi Lambda untuk transcoding batch video yang disimpan dalam bucket sumber S3. Fungsi Lambda memanggil MediaConvert untuk mentranskode video. Output untuk setiap video di bucket sumber S3 adalah sebagai berikut:

- Streaming bitrate adaptif HTTP Live Streaming (HLS) untuk pemutaran pada perangkat dengan berbagai ukuran dan bandwidth yang bervariasi
- File video MP4
- Gambar thumbnail dikumpulkan pada interval

Topik

- Prasyarat
- Langkah 1: Buat bucket S3 untuk file media keluaran
- Langkah 2: Buat peran IAM untuk MediaConvert
- Langkah 3: Buat peran IAM untuk fungsi Lambda Anda
- Langkah 4: Buat fungsi Lambda untuk transcoding video
- Langkah 5: Konfigurasikan Inventaris Amazon S3 untuk bucket sumber S3 Anda
- Langkah 6: Buat peran IAM untuk Operasi Batch S3
- Langkah 7: Buat dan jalankan pekerjaan Operasi Batch S3
- Langkah 8: Memeriksa file media output dari bucket tujuan S3 Anda
- Langkah 9: Membersihkan
- Langkah selanjutnya

Prasyarat

Sebelum Anda memulai tutorial ini, Anda harus memiliki bucket sumber Amazon S3 (misalnya,**`tutorial-bucket-1`**) dengan video yang akan ditranskode sudah disimpan di dalamnya.

Anda dapat memberi ember nama lain jika Anda mau. Untuk informasi selengkapnya tentang nama bucket di Amazon S3, lihat. Peraturan penamaan bucket

Untuk bucket sumber S3, pertahankan pengaturan yang terkait dengan pengaturan Blokir Akses Publik untuk bucket ini disetel ke default (Blokir semua akses publik diaktifkan). Untuk informasi selengkapnya, lihat Membuat bucket.

Untuk informasi selengkapnya tentang mengunggah video ke bucket sumber S3, lihat. Mengunggah Objek Jika Anda mengunggah banyak file video besar ke S3, Anda mungkin ingin menggunakan Amazon S3 Transfer Acceleration untuk mengonfigurasi transfer file yang cepat dan aman. Transfer Acceleration dapat mempercepat pengunggahan video ke bucket S3 Anda untuk transfer jarak jauh video yang lebih besar. Untuk informasi selengkapnya, lihat Mengonfigurasi transfer file yang cepat dan aman menggunakan Amazon S3 Transfer Acceleration.

Langkah 1: Buat bucket S3 untuk file media keluaran

Pada langkah ini, Anda membuat bucket tujuan S3 untuk menyimpan file media keluaran yang dikonversi. Anda juga membuat konfigurasi Cross Origin Resource Sharing (CORS) untuk memungkinkan akses lintas asal ke file media transkode yang disimpan di bucket tujuan S3 Anda.

Sublangkah

- Buat ember untuk file media keluaran
- Tambahkan konfigurasi CORS ke bucket keluaran S3

Buat ember untuk file media keluaran

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih Buat bucket.
4. Untuk nama Bucket, masukkan nama untuk bucket Anda (misalnya,**`tutorial-bucket-2`**).
5. Untuk Wilayah, pilih Wilayah AWS tempat Anda ingin ember berada.

6. Untuk memastikan akses publik ke file media keluaran Anda, dalam pengaturan Blokir Akses Publik untuk bucket ini, hapus Blokir semua akses publik.

> **Warning**
>
> Sebelum Anda menyelesaikan langkah ini, tinjau Melakukan blok akses publik ke penyimpanan Amazon S3 Anda untuk memastikan bahwa Anda telah memahami dan menerima risiko yang terkait dengan mengizinkan akses publik. Saat Anda mematikan pengaturan Blokir Akses Publik untuk membuat bucket Anda publik, siapa pun di internet dapat mengakses bucket Anda. Kami sarankan agar Anda memblokir semua akses publik ke bucket Anda.
>
> Jika Anda tidak ingin menghapus pengaturan Blokir Akses Publik, Anda dapat menggunakan Amazon CloudFront untuk mengirimkan file media yang ditranskode ke pemirsa (pengguna akhir). Untuk informasi selengkapnya, lihat Tutorial: Hosting video streaming sesuai permintaan dengan Amazon S3, Amazon, dan CloudFront Amazon Route 53.

7. Pilih kotak centang di sebelah Saya mengakui bahwa pengaturan saat ini dapat mengakibatkan bucket ini dan objek dalam menjadi publik.
8. Pertahankan pengaturan yang tersisa disetel ke default.
9. Pilih Buat bucket.

Tambahkan konfigurasi CORS ke bucket keluaran S3

Konfigurasi JSON CORS mendefinisikan cara untuk aplikasi web klien (pemutar video dalam konteks ini) yang dimuat dalam satu domain untuk memutar file media keluaran transkode dalam domain yang berbeda.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dalam daftar Bucket, pilih nama bucket yang Anda buat sebelumnya (misalnya,**`tutorial-bucket-2`**).
4. Pilih tab Izin.
5. Di bagian Berbagi sumber daya lintas asal (CORS), pilih Edit.
6. Di kotak teks konfigurasi CORS, salin dan tempel konfigurasi CORS berikut.

Konfigurasi CORS harus dalam format JSON. Dalam contoh ini, `AllowedOrigins` atribut menggunakan karakter wildcard (`*`) untuk menentukan semua asal. Jika Anda tahu asal spesifik Anda, Anda dapat membatasi `AllowedOrigins` atribut ke URL pemain spesifik Anda. Untuk informasi selengkapnya tentang mengonfigurasi atribut ini dan atribut lainnya, lihatKonfigurasi CORS.

```
[
    {
        "AllowedOrigins": [
            "*"
        ],
        "AllowedMethods": [
            "GET"
        ],
        "AllowedHeaders": [
            "*"
        ],
        "ExposeHeaders": []

    }
]
```

7. Pilih Simpan perubahan.

Langkah 2: Buat peran IAM untuk MediaConvert

Untuk digunakan AWS Elemental MediaConvert untuk mentranskode video input yang disimpan di bucket S3, Anda harus memiliki peran layanan AWS Identity and Access Management (IAM) untuk memberikan MediaConvert izin membaca dan menulis file video dari dan ke bucket sumber dan tujuan S3 Anda. Saat Anda menjalankan pekerjaan transcoding, MediaConvert konsol menggunakan peran ini.

Untuk membuat peran IAM untuk MediaConvert

1. Buat peran IAM dengan nama peran yang Anda pilih (misalnya,**`tutorial-mediaconvert-role`**). Untuk membuat peran ini, ikuti langkah-langkah di Buat MediaConvert peran Anda di IAM (konsol) di Panduan AWS Elemental MediaConvert Pengguna.
2. Setelah Anda membuat peran IAM untuk MediaConvert, dalam daftar Peran, pilih nama peran MediaConvert yang Anda buat (misalnya,**`tutorial-mediaconvert-role`**).

3. Pada halaman Ringkasan, salin ARN Peran (yang dimulai dengan`arn:aws:iam::`), dan simpan ARN untuk digunakan nanti.

   Untuk informasi tentang ARN, lihat Amazon Resource Name (ARN) di Referensi Umum AWS .

Langkah 3: Buat peran IAM untuk fungsi Lambda Anda

Untuk mentranskode video batch dengan dan Operasi Batch MediaConvert S3, Anda menggunakan fungsi Lambda untuk menghubungkan kedua layanan ini untuk mengonversi video. Fungsi Lambda ini harus memiliki peran IAM yang memberikan izin fungsi Lambda untuk MediaConvert mengakses dan Operasi Batch S3.

Sublangkah

- Buat peran IAM untuk fungsi Lambda Anda
- Sematkan kebijakan inline untuk peran IAM fungsi Lambda Anda

Buat peran IAM untuk fungsi Lambda Anda

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi kiri, pilih Peran, lalu pilih Buat peran.
3. Pilih jenis peran AWS layanan, lalu di bawah Kasus penggunaan umum, pilih Lambda.
4. Pilih Berikutnya: Izin.
5. Pada halaman Lampirkan kebijakan izin, masukkan **`AWSLambdaBasicExecutionRole`** kotak Filter kebijakan. Untuk melampirkan kebijakan terkelola AWSLambdaBasicExecutionRoleke peran ini untuk memberikan izin menulis ke CloudWatch Log Amazon, pilih kotak centang di AWSLambdaBasicExecutionRolesebelahnya.
6. Pilih Berikutnya: Tanda.
7. (Opsional) Tambahkan tag ke kebijakan terkelola.
8. Pilih Selanjutnya: Tinjau.
9. Untuk Nama peran, masukkan **`tutorial-lambda-transcode-role`**.
10. Pilih Buat peran.

Sematkan kebijakan inline untuk peran IAM fungsi Lambda Anda

Untuk memberikan izin ke MediaConvert sumber daya yang diperlukan agar fungsi Lambda dapat dijalankan, Anda harus menggunakan kebijakan sebaris.

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.
3. Dalam daftar Peran, pilih nama peran IAM yang Anda buat sebelumnya untuk fungsi Lambda Anda (misalnya**`tutorial-lambda-transcode-role`**,).
4. Pilih tab Izin.
5. Pilih Menambahkan kebijakan inline.
6. Pilih tab JSON, lalu salin dan tempel kebijakan JSON berikut.

   Dalam kebijakan JSON, ganti contoh nilai ARN dengan peran ARN `Resource` dari peran IAM MediaConvert untuk yang Anda buat di Langkah 2 (misalnya,). **`tutorial-mediaconvert-role`**

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "Logging"
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::111122223333:role/tutorial-mediaconvert-role"
            ],
            "Effect": "Allow",
            "Sid": "PassRole"
```

```
        },
        {
            "Action": [
                "mediaconvert:*"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "MediaConvertService"
        },
        {
            "Action": [
                "s3:*"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "S3Service"
        }
    ]
}
```

7. Pilih Tinjau Kebijakan.
8. Untuk Nama, masukkan **`tutorial-lambda-policy`**.
9. Pilih Buat Kebijakan.

   Setelah Anda membuat kebijakan sebaris, kebijakan tersebut secara otomatis disematkan dalam peran IAM fungsi Lambda Anda.

Langkah 4: Buat fungsi Lambda untuk transcoding video

Di bagian tutorial ini, Anda membangun fungsi Lambda menggunakan SDK untuk Python untuk diintegrasikan dengan Operasi Batch S3 dan. MediaConvert Untuk memulai transcoding video yang sudah disimpan di bucket sumber S3, Anda menjalankan tugas Operasi Batch S3 yang secara langsung memanggil fungsi Lambda untuk setiap video di bucket sumber S3. Kemudian, fungsi Lambda mengirimkan pekerjaan transcoding untuk setiap video. MediaConvert

Sublangkah

- Tulis kode fungsi Lambda dan buat paket penerapan

- Buat fungsi Lambda dengan peran eksekusi (konsol)
- Terapkan fungsi Lambda Anda dengan arsip file.zip dan konfigurasikan fungsi Lambda (konsol)

Tulis kode fungsi Lambda dan buat paket penerapan

1. Di mesin lokal Anda, buat folder bernama`batch-transcode`.
2. Di `batch-transcode` folder, buat file dengan pengaturan pekerjaan JSON. Misalnya, Anda dapat menggunakan pengaturan yang disediakan di bagian ini, dan beri nama file`job.json`.

   Sebuah `job.json` file menentukan berikut:

   - File mana yang akan ditranskode
   - Bagaimana Anda ingin mentranskode video input Anda
   - File media keluaran apa yang ingin Anda buat
   - Apa yang harus diberi nama file yang ditranskode
   - Tempat menyimpan file yang ditranskode
   - Fitur canggih mana yang akan diterapkan, dan sebagainya

   Dalam tutorial ini, kita menggunakan `job.json` file berikut untuk membuat output berikut untuk setiap video di bucket sumber S3:

   - Streaming bitrate adaptif HTTP Live Streaming (HLS) untuk pemutaran di beberapa perangkat dengan ukuran berbeda dan bandwidth yang bervariasi
   - File video MP4
   - Gambar thumbnail dikumpulkan pada interval

   `job.json`File contoh ini menggunakan Quality-Defined Variable Bitrate (QVBR) untuk mengoptimalkan kualitas video. Output HLS sesuai dengan Apple (audio tidak dicampur dari video, durasi segmen 6 detik, dan kualitas video yang dioptimalkan melalui QVBR otomatis).

   Jika Anda tidak ingin menggunakan contoh pengaturan yang disediakan di sini, Anda dapat membuat `job.json` spesifikasi berdasarkan kasus penggunaan Anda. Untuk memastikan konsistensi di seluruh output Anda, pastikan file input Anda memiliki konfigurasi video dan audio yang serupa. Untuk file input apa pun dengan konfigurasi video dan audio yang berbeda, buat otomatisasi terpisah (`job.json`pengaturan unik). Untuk informasi selengkapnya, lihat Contoh

pengaturan AWS Elemental MediaConvert pekerjaan di JSON di Panduan AWS Elemental MediaConvert Pengguna.

```
{
  "OutputGroups": [
    {
      "CustomName": "HLS",
      "Name": "Apple HLS",
      "Outputs": [
        {
          "ContainerSettings": {
            "Container": "M3U8",
            "M3u8Settings": {
              "AudioFramesPerPes": 4,
              "PcrControl": "PCR_EVERY_PES_PACKET",
              "PmtPid": 480,
              "PrivateMetadataPid": 503,
              "ProgramNumber": 1,
              "PatInterval": 0,
              "PmtInterval": 0,
              "TimedMetadata": "NONE",
              "VideoPid": 481,
              "AudioPids": [
                482,
                483,
                484,
                485,
                486,
                487,
                488,
                489,
                490,
                491,
                492
              ]
            }
          },
          "VideoDescription": {
            "Width": 640,
            "ScalingBehavior": "DEFAULT",
            "Height": 360,
            "TimecodeInsertion": "DISABLED",
            "AntiAlias": "ENABLED",
```

```
            "Sharpness": 50,
            "CodecSettings": {
              "Codec": "H_264",
              "H264Settings": {
                "InterlaceMode": "PROGRESSIVE",
                "NumberReferenceFrames": 3,
                "Syntax": "DEFAULT",
                "Softness": 0,
                "GopClosedCadence": 1,
                "GopSize": 2,
                "Slices": 1,
                "GopBReference": "DISABLED",
                "MaxBitrate": 1200000,
                "SlowPal": "DISABLED",
                "SpatialAdaptiveQuantization": "ENABLED",
                "TemporalAdaptiveQuantization": "ENABLED",
                "FlickerAdaptiveQuantization": "DISABLED",
                "EntropyEncoding": "CABAC",
                "FramerateControl": "INITIALIZE_FROM_SOURCE",
                "RateControlMode": "QVBR",
                "CodecProfile": "MAIN",
                "Telecine": "NONE",
                "MinIInterval": 0,
                "AdaptiveQuantization": "HIGH",
                "CodecLevel": "AUTO",
                "FieldEncoding": "PAFF",
                "SceneChangeDetect": "TRANSITION_DETECTION",
                "QualityTuningLevel": "SINGLE_PASS_HQ",
                "FramerateConversionAlgorithm": "DUPLICATE_DROP",
                "UnregisteredSeiTimecode": "DISABLED",
                "GopSizeUnits": "SECONDS",
                "ParControl": "INITIALIZE_FROM_SOURCE",
                "NumberBFramesBetweenReferenceFrames": 2,
                "RepeatPps": "DISABLED"
              }
            },
            "AfdSignaling": "NONE",
            "DropFrameTimecode": "ENABLED",
            "RespondToAfd": "NONE",
            "ColorMetadata": "INSERT"
          },
          "OutputSettings": {
            "HlsSettings": {
              "AudioGroupId": "program_audio",
```

```
            "AudioRenditionSets": "program_audio",
            "SegmentModifier": "$dt$",
            "IFrameOnlyManifest": "EXCLUDE"
          }
        },
        "NameModifier": "_360"
      },
      {
        "ContainerSettings": {
          "Container": "M3U8",
          "M3u8Settings": {
            "AudioFramesPerPes": 4,
            "PcrControl": "PCR_EVERY_PES_PACKET",
            "PmtPid": 480,
            "PrivateMetadataPid": 503,
            "ProgramNumber": 1,
            "PatInterval": 0,
            "PmtInterval": 0,
            "TimedMetadata": "NONE",
            "TimedMetadataPid": 502,
            "VideoPid": 481,
            "AudioPids": [
              482,
              483,
              484,
              485,
              486,
              487,
              488,
              489,
              490,
              491,
              492
            ]
          }
        },
        "VideoDescription": {
          "Width": 960,
          "ScalingBehavior": "DEFAULT",
          "Height": 540,
          "TimecodeInsertion": "DISABLED",
          "AntiAlias": "ENABLED",
          "Sharpness": 50,
          "CodecSettings": {
```

```
                "Codec": "H_264",
                "H264Settings": {
                  "InterlaceMode": "PROGRESSIVE",
                  "NumberReferenceFrames": 3,
                  "Syntax": "DEFAULT",
                  "Softness": 0,
                  "GopClosedCadence": 1,
                  "GopSize": 2,
                  "Slices": 1,
                  "GopBReference": "DISABLED",
                  "MaxBitrate": 3500000,
                  "SlowPal": "DISABLED",
                  "SpatialAdaptiveQuantization": "ENABLED",
                  "TemporalAdaptiveQuantization": "ENABLED",
                  "FlickerAdaptiveQuantization": "DISABLED",
                  "EntropyEncoding": "CABAC",
                  "FramerateControl": "INITIALIZE_FROM_SOURCE",
                  "RateControlMode": "QVBR",
                  "CodecProfile": "MAIN",
                  "Telecine": "NONE",
                  "MinIInterval": 0,
                  "AdaptiveQuantization": "HIGH",
                  "CodecLevel": "AUTO",
                  "FieldEncoding": "PAFF",
                  "SceneChangeDetect": "TRANSITION_DETECTION",
                  "QualityTuningLevel": "SINGLE_PASS_HQ",
                  "FramerateConversionAlgorithm": "DUPLICATE_DROP",
                  "UnregisteredSeiTimecode": "DISABLED",
                  "GopSizeUnits": "SECONDS",
                  "ParControl": "INITIALIZE_FROM_SOURCE",
                  "NumberBFramesBetweenReferenceFrames": 2,
                  "RepeatPps": "DISABLED"
                }
              },
              "AfdSignaling": "NONE",
              "DropFrameTimecode": "ENABLED",
              "RespondToAfd": "NONE",
              "ColorMetadata": "INSERT"
            },
            "OutputSettings": {
              "HlsSettings": {
                "AudioGroupId": "program_audio",
                "AudioRenditionSets": "program_audio",
                "SegmentModifier": "$dt$",
```

```
            "IFrameOnlyManifest": "EXCLUDE"
          }
        },
        "NameModifier": "_540"
      },
      {
        "ContainerSettings": {
          "Container": "M3U8",
          "M3u8Settings": {
            "AudioFramesPerPes": 4,
            "PcrControl": "PCR_EVERY_PES_PACKET",
            "PmtPid": 480,
            "PrivateMetadataPid": 503,
            "ProgramNumber": 1,
            "PatInterval": 0,
            "PmtInterval": 0,
            "TimedMetadata": "NONE",
            "VideoPid": 481,
            "AudioPids": [
              482,
              483,
              484,
              485,
              486,
              487,
              488,
              489,
              490,
              491,
              492
            ]
          }
        },
        "VideoDescription": {
          "Width": 1280,
          "ScalingBehavior": "DEFAULT",
          "Height": 720,
          "TimecodeInsertion": "DISABLED",
          "AntiAlias": "ENABLED",
          "Sharpness": 50,
          "CodecSettings": {
            "Codec": "H_264",
            "H264Settings": {
              "InterlaceMode": "PROGRESSIVE",
```

```
                "NumberReferenceFrames": 3,
                "Syntax": "DEFAULT",
                "Softness": 0,
                "GopClosedCadence": 1,
                "GopSize": 2,
                "Slices": 1,
                "GopBReference": "DISABLED",
                "MaxBitrate": 5000000,
                "SlowPal": "DISABLED",
                "SpatialAdaptiveQuantization": "ENABLED",
                "TemporalAdaptiveQuantization": "ENABLED",
                "FlickerAdaptiveQuantization": "DISABLED",
                "EntropyEncoding": "CABAC",
                "FramerateControl": "INITIALIZE_FROM_SOURCE",
                "RateControlMode": "QVBR",
                "CodecProfile": "MAIN",
                "Telecine": "NONE",
                "MinIInterval": 0,
                "AdaptiveQuantization": "HIGH",
                "CodecLevel": "AUTO",
                "FieldEncoding": "PAFF",
                "SceneChangeDetect": "TRANSITION_DETECTION",
                "QualityTuningLevel": "SINGLE_PASS_HQ",
                "FramerateConversionAlgorithm": "DUPLICATE_DROP",
                "UnregisteredSeiTimecode": "DISABLED",
                "GopSizeUnits": "SECONDS",
                "ParControl": "INITIALIZE_FROM_SOURCE",
                "NumberBFramesBetweenReferenceFrames": 2,
                "RepeatPps": "DISABLED"
              }
            },
            "AfdSignaling": "NONE",
            "DropFrameTimecode": "ENABLED",
            "RespondToAfd": "NONE",
            "ColorMetadata": "INSERT"
          },
          "OutputSettings": {
            "HlsSettings": {
              "AudioGroupId": "program_audio",
              "AudioRenditionSets": "program_audio",
              "SegmentModifier": "$dt$",
              "IFrameOnlyManifest": "EXCLUDE"
            }
          },
```

```
          "NameModifier": "_720"
        },
        {
          "ContainerSettings": {
            "Container": "M3U8",
            "M3u8Settings": {}
          },
          "AudioDescriptions": [
            {
              "AudioSourceName": "Audio Selector 1",
              "CodecSettings": {
                "Codec": "AAC",
                "AacSettings": {
                  "Bitrate": 96000,
                  "CodingMode": "CODING_MODE_2_0",
                  "SampleRate": 48000
                }
              }
            }
          ],
          "OutputSettings": {
            "HlsSettings": {
              "AudioGroupId": "program_audio",
              "AudioTrackType": "ALTERNATE_AUDIO_AUTO_SELECT_DEFAULT"
            }
          },
          "NameModifier": "_audio"
        }
      ],
      "OutputGroupSettings": {
        "Type": "HLS_GROUP_SETTINGS",
        "HlsGroupSettings": {
          "ManifestDurationFormat": "INTEGER",
          "SegmentLength": 6,
          "TimedMetadataId3Period": 10,
          "CaptionLanguageSetting": "OMIT",
          "Destination": "s3://EXAMPLE-BUCKET/HLS/",
          "DestinationSettings": {
            "S3Settings": {
              "AccessControl": {
                "CannedAcl": "PUBLIC_READ"
              }
            }
          },
```

```
          "TimedMetadataId3Frame": "PRIV",
          "CodecSpecification": "RFC_4281",
          "OutputSelection": "MANIFESTS_AND_SEGMENTS",
          "ProgramDateTimePeriod": 600,
          "MinSegmentLength": 0,
          "DirectoryStructure": "SINGLE_DIRECTORY",
          "ProgramDateTime": "EXCLUDE",
          "SegmentControl": "SEGMENTED_FILES",
          "ManifestCompression": "NONE",
          "ClientCache": "ENABLED",
          "StreamInfResolution": "INCLUDE"
        }
      }
    },
    {
      "CustomName": "MP4",
      "Name": "File Group",
      "Outputs": [
        {
          "ContainerSettings": {
            "Container": "MP4",
            "Mp4Settings": {
              "CslgAtom": "INCLUDE",
              "FreeSpaceBox": "EXCLUDE",
              "MoovPlacement": "PROGRESSIVE_DOWNLOAD"
            }
          },
          "VideoDescription": {
            "Width": 1280,
            "ScalingBehavior": "DEFAULT",
            "Height": 720,
            "TimecodeInsertion": "DISABLED",
            "AntiAlias": "ENABLED",
            "Sharpness": 100,
            "CodecSettings": {
              "Codec": "H_264",
              "H264Settings": {
                "InterlaceMode": "PROGRESSIVE",
                "ParNumerator": 1,
                "NumberReferenceFrames": 3,
                "Syntax": "DEFAULT",
                "Softness": 0,
                "GopClosedCadence": 1,
                "HrdBufferInitialFillPercentage": 90,
```

```
                "GopSize": 2,
                "Slices": 2,
                "GopBReference": "ENABLED",
                "HrdBufferSize": 10000000,
                "MaxBitrate": 5000000,
                "ParDenominator": 1,
                "EntropyEncoding": "CABAC",
                "RateControlMode": "QVBR",
                "CodecProfile": "HIGH",
                "MinIInterval": 0,
                "AdaptiveQuantization": "AUTO",
                "CodecLevel": "AUTO",
                "FieldEncoding": "PAFF",
                "SceneChangeDetect": "ENABLED",
                "QualityTuningLevel": "SINGLE_PASS_HQ",
                "UnregisteredSeiTimecode": "DISABLED",
                "GopSizeUnits": "SECONDS",
                "ParControl": "SPECIFIED",
                "NumberBFramesBetweenReferenceFrames": 3,
                "RepeatPps": "DISABLED",
                "DynamicSubGop": "ADAPTIVE"
              }
            },
            "AfdSignaling": "NONE",
            "DropFrameTimecode": "ENABLED",
            "RespondToAfd": "NONE",
            "ColorMetadata": "INSERT"
          },
          "AudioDescriptions": [
            {
              "AudioTypeControl": "FOLLOW_INPUT",
              "AudioSourceName": "Audio Selector 1",
              "CodecSettings": {
                "Codec": "AAC",
                "AacSettings": {
                  "AudioDescriptionBroadcasterMix": "NORMAL",
                  "Bitrate": 160000,
                  "RateControlMode": "CBR",
                  "CodecProfile": "LC",
                  "CodingMode": "CODING_MODE_2_0",
                  "RawFormat": "NONE",
                  "SampleRate": 48000,
                  "Specification": "MPEG4"
                }
```

```
              },
              "LanguageCodeControl": "FOLLOW_INPUT",
              "AudioType": 0
            }
          ]
        }
      ],
      "OutputGroupSettings": {
        "Type": "FILE_GROUP_SETTINGS",
        "FileGroupSettings": {
          "Destination": "s3://EXAMPLE-BUCKET/MP4/",
          "DestinationSettings": {
            "S3Settings": {
              "AccessControl": {
                "CannedAcl": "PUBLIC_READ"
              }
            }
          }
        }
      }
    },
    {
      "CustomName": "Thumbnails",
      "Name": "File Group",
      "Outputs": [
        {
          "ContainerSettings": {
            "Container": "RAW"
          },
          "VideoDescription": {
            "Width": 1280,
            "ScalingBehavior": "DEFAULT",
            "Height": 720,
            "TimecodeInsertion": "DISABLED",
            "AntiAlias": "ENABLED",
            "Sharpness": 50,
            "CodecSettings": {
              "Codec": "FRAME_CAPTURE",
              "FrameCaptureSettings": {
                "FramerateNumerator": 1,
                "FramerateDenominator": 5,
                "MaxCaptures": 500,
                "Quality": 80
              }
```

```
            },
            "AfdSignaling": "NONE",
            "DropFrameTimecode": "ENABLED",
            "RespondToAfd": "NONE",
            "ColorMetadata": "INSERT"
          }
        }
      ],
      "OutputGroupSettings": {
        "Type": "FILE_GROUP_SETTINGS",
        "FileGroupSettings": {
          "Destination": "s3://EXAMPLE-BUCKET/Thumbnails/",
          "DestinationSettings": {
            "S3Settings": {
              "AccessControl": {
                "CannedAcl": "PUBLIC_READ"
              }
            }
          }
        }
      }
    }
  ],
  "AdAvailOffset": 0,
  "Inputs": [
    {
      "AudioSelectors": {
        "Audio Selector 1": {
          "Offset": 0,
          "DefaultSelection": "DEFAULT",
          "ProgramSelection": 1
        }
      },
      "VideoSelector": {
        "ColorSpace": "FOLLOW"
      },
      "FilterEnable": "AUTO",
      "PsiControl": "USE_PSI",
      "FilterStrength": 0,
      "DeblockFilter": "DISABLED",
      "DenoiseFilter": "DISABLED",
      "TimecodeSource": "EMBEDDED",
      "FileInput": "s3://EXAMPLE-INPUT-BUCKET/input.mp4"
    }
```

```
  ]
}
```

3. Di `batch-transcode` folder, buat file dengan fungsi Lambda. Anda dapat menggunakan contoh Python berikut dan memberi nama file. `convert.py`

   Operasi Batch S3 mengirimkan data tugas tertentu ke fungsi Lambda dan memerlukan data hasil kembali. Untuk contoh permintaan dan respons untuk fungsi Lambda, informasi tentang kode respons dan hasil, dan contoh fungsi Lambda untuk Operasi Batch S3, lihat. Memanggil fungsi AWS Lambda

```
import json
import os
from urllib.parse import urlparse
import uuid
import boto3

"""
When you run an S3 Batch Operations job, your job
invokes this Lambda function. Specifically, the Lambda function is
invoked on each video object listed in the manifest that you specify
for the S3 Batch Operations job in Step 5.

Input parameter "event": The S3 Batch Operations event as a request
                         for the Lambda function.

Input parameter "context": Context about the event.

Output: A result structure that Amazon S3 uses to interpret the result
        of the operation. It is a job response returned back to S3 Batch
 Operations.
"""
def handler(event, context):

    invocation_schema_version = event['invocationSchemaVersion']
    invocation_id = event['invocationId']
    task_id = event['tasks'][0]['taskId']

    source_s3_key = event['tasks'][0]['s3Key']
    source_s3_bucket = event['tasks'][0]['s3BucketArn'].split(':::')[-1]
    source_s3 = 's3://' + source_s3_bucket + '/' + source_s3_key

    result_list = []
```

```
    result_code = 'Succeeded'
    result_string = 'The input video object was converted successfully.'

    # The type of output group determines which media players can play
    # the files transcoded by MediaConvert.
    # For more information, see Creating outputs with AWS Elemental MediaConvert.
    output_group_type_dict = {
        'HLS_GROUP_SETTINGS': 'HlsGroupSettings',
        'FILE_GROUP_SETTINGS': 'FileGroupSettings',
        'CMAF_GROUP_SETTINGS': 'CmafGroupSettings',
        'DASH_ISO_GROUP_SETTINGS': 'DashIsoGroupSettings',
        'MS_SMOOTH_GROUP_SETTINGS': 'MsSmoothGroupSettings'
    }

    try:
        job_name = 'Default'
        with open('job.json') as file:
            job_settings = json.load(file)

        job_settings['Inputs'][0]['FileInput'] = source_s3

        # The path of each output video is constructed based on the values of
        # the attributes in each object of OutputGroups in the job.json file.
        destination_s3 = 's3://{0}/{1}/{2}' \
            .format(os.environ['DestinationBucket'],
                    os.path.splitext(os.path.basename(source_s3_key))[0],
                    os.path.splitext(os.path.basename(job_name))[0])

        for output_group in job_settings['OutputGroups']:
            output_group_type = output_group['OutputGroupSettings']['Type']
            if output_group_type in output_group_type_dict.keys():
                output_group_type = output_group_type_dict[output_group_type]
                output_group['OutputGroupSettings'][output_group_type]
['Destination'] = \
                    "{0}{1}".format(destination_s3,
                                    urlparse(output_group['OutputGroupSettings']
[output_group_type]['Destination']).path)
            else:
                raise ValueError("Exception: Unknown Output Group Type {}."
                                 .format(output_group_type))

        job_metadata_dict = {
            'assetID': str(uuid.uuid4()),
            'application': os.environ['Application'],
```

```
            'input': source_s3,
            'settings': job_name
        }

        region = os.environ['AWS_DEFAULT_REGION']
        endpoints = boto3.client('mediaconvert', region_name=region) \
            .describe_endpoints()
        client = boto3.client('mediaconvert', region_name=region,
                              endpoint_url=endpoints['Endpoints'][0]['Url'],
                              verify=False)

        try:
            client.create_job(Role=os.environ['MediaConvertRole'],
                              UserMetadata=job_metadata_dict,
                              Settings=job_settings)
        # You can customize error handling based on different error codes that
        # MediaConvert can return.
        # For more information, see MediaConvert error codes.
        # When the result_code is TemporaryFailure, S3 Batch Operations retries
        # the task before the job is completed. If this is the final retry,
        # the error message is included in the final report.
        except Exception as error:
            result_code = 'TemporaryFailure'
            raise

    except Exception as error:
        if result_code != 'TemporaryFailure':
            result_code = 'PermanentFailure'
        result_string = str(error)

    finally:
        result_list.append({
            'taskId': task_id,
            'resultCode': result_code,
            'resultString': result_string,
        })

    return {
        'invocationSchemaVersion': invocation_schema_version,
        'treatMissingKeyAs': 'PermanentFailure',
        'invocationId': invocation_id,
        'results': result_list
    }
```

4. Untuk membuat paket penyebaran dengan `convert.py` dan `job.json` sebagai `.zip` file bernama`lambda.zip`, di terminal lokal Anda, buka `batch-transcode` folder yang Anda buat sebelumnya, dan jalankan perintah berikut.

   Untuk pengguna macOS, jalankan perintah berikut:

   ```
   zip -r lambda.zip convert.py job.json
   ```

   Untuk pengguna Windows, jalankan perintah berikut:

   ```
   powershell Compress-Archive convert.py lambda.zip
   ```

   ```
   powershell Compress-Archive -update job.json lambda.zip
   ```

Buat fungsi Lambda dengan peran eksekusi (konsol)

1. Buka AWS Lambda konsol di https://console.aws.amazon.com/lambda/.
2. Di panel navigasi kiri, pilih Fungsi.
3. Pilih Buat fungsi.
4. Pilih Penulis dari scratch.
5. Di bagian Informasi dasar, lakukan hal berikut:
   a. Untuk Nama fungsi, masukkan **`tutorial-lambda-convert`**.
   b. Untuk Runtime, pilih Python 3.8 atau versi Python yang lebih baru.
6. Pilih Ubah peran eksekusi default, dan di bawah Peran eksekusi, pilih Gunakan peran yang ada.
7. Di bawah Peran yang ada, pilih nama peran IAM yang Anda buat untuk fungsi Lambda di Langkah 3 (misalnya**`tutorial-lambda-transcode-role`**,).
8. Untuk pengaturan yang tersisa, pertahankan defaultnya.
9. Pilih Buat fungsi.

Terapkan fungsi Lambda Anda dengan arsip file.zip dan konfigurasikan fungsi Lambda (konsol)

1. Di bagian Sumber Kode halaman untuk fungsi Lambda yang Anda buat (misalnya,**`tutorial-lambda-convert`**), pilih Unggah dari lalu file.zip.
2. Pilih Unggah untuk memilih `.zip` file lokal Anda.
3. Pilih `lambda.zip` file yang Anda buat sebelumnya, dan pilih Buka.
4. Pilih Simpan.
5. Di bagian Pengaturan waktu proses, pilih Edit.
6. Untuk memberi tahu runtime Lambda metode handler mana dalam kode fungsi Lambda Anda yang akan dipanggil, masukkan di bidang Handler. **`convert.handler`**

   Ketika Anda mengkonfigurasi fungsi dalam Python, nilai pengaturan handler adalah nama file dan nama modul handler, dipisahkan oleh titik (). `.` Misalnya, `convert.handler` memanggil `handler` metode yang didefinisikan dalam `convert.py` file.
7. Pilih Simpan.
8. Pada halaman fungsi Lambda Anda, pilih tab Konfigurasi. Di panel navigasi kiri pada tab Konfigurasi, pilih Variabel lingkungan, lalu pilih Edit.
9. Pilih Tambahkan variabel lingkungan. Kemudian, masukkan Kunci dan Nilai yang ditentukan untuk masing-masing variabel lingkungan berikut:

   - Kunci: **`DestinationBucket`** Nilai: **`tutorial-bucket-2`**

     Nilai ini adalah bucket S3 untuk file media keluaran yang Anda buat di Langkah 1.
   - Kunci: **`MediaConvertRole`** Nilai: **`arn:aws:iam::`*`111122223333`*`:role/tutorial-mediaconvert-role`**

     Nilai ini adalah ARN dari peran IAM untuk MediaConvert yang Anda buat di Langkah 2. Pastikan untuk mengganti ARN ini dengan ARN sebenarnya dari peran IAM Anda.
   - Kunci: **`Application`** Nilai: **`Batch-Transcoding`**

     Nilai ini adalah nama aplikasi.
10. Pilih Simpan.
11. (Opsional) Pada tab Konfigurasi, di bagian Konfigurasi umum pada panel navigasi kiri, pilih Edit. Di bidang Timeout, masukkan **`2`** min **`0`** sec. Lalu, pilih Simpan.

Timeout adalah jumlah waktu yang memungkinkan Lambda menjalankan fungsi untuk pemanggilan sebelum menghentikannya. Default-nya adalah 3 detik. Harga didasarkan pada jumlah memori yang dikonfigurasi dan jumlah waktu kode Anda berjalan. Untuk informasi selengkapnya, lihat harga AWS Lambda.

Langkah 5: Konfigurasikan Inventaris Amazon S3 untuk bucket sumber S3 Anda

Setelah mengatur fungsi transcoding Lambda, buat pekerjaan Operasi Batch S3 untuk mentranskode satu set video. Pertama, Anda memerlukan daftar objek video masukan yang Anda inginkan Operasi Batch S3 untuk menjalankan tindakan transcoding yang ditentukan. Untuk mendapatkan daftar objek video masukan, Anda dapat membuat laporan Inventaris S3 untuk bucket sumber S3 Anda (misalnya,**`tutorial-bucket-1`**).

Sublangkah

- Membuat dan mengonfigurasi bucket untuk laporan Inventaris S3 untuk video masukan
- Konfigurasikan Inventaris Amazon S3 untuk bucket sumber video S3 Anda
- Periksa laporan inventaris untuk bucket sumber video S3 Anda

Membuat dan mengonfigurasi bucket untuk laporan Inventaris S3 untuk video masukan

Untuk menyimpan laporan Inventaris S3 yang mencantumkan objek bucket sumber S3, buat bucket tujuan Inventaris S3, lalu konfigurasikan kebijakan bucket untuk bucket untuk menulis file inventaris ke bucket sumber S3.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih Buat bucket.
4. Untuk nama Bucket, masukkan nama untuk bucket Anda (misalnya,**`tutorial-bucket-3`**).
5. Untuk Wilayah AWS, pilih di Wilayah AWS mana Anda ingin ember berada.

   Bucket tujuan inventaris harus Wilayah AWS sama dengan bucket sumber tempat Anda menyiapkan Inventaris S3. Bucket tujuan inventaris bisa berbeda Akun AWS.

6. Di pengaturan Blokir Akses Publik untuk bucket ini, pertahankan pengaturan default (Blokir semua akses publik diaktifkan).
7. Untuk pengaturan yang tersisa, pertahankan defaultnya.
8. Pilih Buat bucket.
9. Dalam daftar Bucket, pilih nama bucket yang baru saja Anda buat (misalnya,**`tutorial-bucket-3`**).
10. Untuk memberikan izin Amazon S3 untuk menulis data laporan inventaris ke bucket tujuan Inventaris S3, pilih tab Izin.
11. Gulir ke bawah ke bagian Kebijakan Bucket, lalu pilih Edit. Halaman kebijakan Bucket terbuka.
12. Untuk memberikan izin untuk Inventaris S3, di bidang Kebijakan, tempel kebijakan bucket berikut.

    Ganti tiga nilai contoh dengan nilai-nilai berikut:

    - Nama bucket yang Anda buat untuk menyimpan laporan inventaris (misalnya,*`tutorial-bucket-3`*).
    - Nama bucket sumber yang menyimpan video input (misalnya,*`tutorial-bucket-1`*).
    - Akun AWS ID yang Anda gunakan untuk membuat bucket sumber video S3 (misalnya,*`111122223333`*.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"InventoryAndAnalyticsExamplePolicy",
      "Effect":"Allow",
      "Principal": {"Service": "s3.amazonaws.com"},
      "Action":"s3:PutObject",
      "Resource":["arn:aws:s3:::tutorial-bucket-3/*"],
      "Condition": {
          "ArnLike": {
              "aws:SourceArn": "arn:aws:s3:::tutorial-bucket-1"
           },
         "StringEquals": {
             "aws:SourceAccount": "111122223333",
             "s3:x-amz-acl": "bucket-owner-full-control"
          }
       }
```

```
    }
  ]
}
```

13. Pilih Simpan perubahan.

Konfigurasikan Inventaris Amazon S3 untuk bucket sumber video S3 Anda

Untuk membuat daftar file datar objek video dan metadata, Anda harus mengonfigurasi S3 Inventory untuk bucket sumber video S3 Anda. Laporan inventaris terjadwal ini dapat menyertakan semua objek dalam bucket atau objek yang dikelompokkan berdasarkan awalan bersama. Dalam tutorial ini, laporan S3 Inventory mencakup semua objek video di bucket sumber S3 Anda.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Untuk mengonfigurasi laporan Inventaris S3 dari video masukan di bucket sumber S3, dalam daftar Bucket, pilih nama bucket sumber S3 (misalnya,). **`tutorial-bucket-1`**
4. Pilih tab Manajemen.
5. Gulir ke bawah ke bagian Konfigurasi inventaris, dan pilih Buat konfigurasi inventaris.
6. Untuk nama konfigurasi Inventaris, masukkan nama (misalnya,**`tutorial-inventory-config`**).
7. Di bawah lingkup Inventaris, pilih Versi saat ini hanya untuk versi Objek dan simpan pengaturan cakupan Inventaris lainnya disetel ke default untuk tutorial ini.
8. Di bagian Laporkan detail, untuk bucket Tujuan, pilih Akun ini.
9. Untuk Tujuan, pilih Jelajahi S3, dan pilih bucket tujuan yang Anda buat sebelumnya untuk menyimpan laporan inventaris (misalnya,**`tutorial-bucket-3`**). Kemudian, pilih Pilih jalur.

   Bucket tujuan inventaris harus Wilayah AWS sama dengan bucket sumber tempat Anda menyiapkan Inventaris S3. Bucket tujuan inventaris bisa berbeda Akun AWS.

   Di bagian bucket Destination, izin bucket Destination ditambahkan ke kebijakan bucket tujuan inventaris, yang memungkinkan Amazon S3 menempatkan data di bucket tujuan inventaris. Untuk informasi selengkapnya, lihat Membuat kebijakan bucket tujuan.
10. Untuk Frekuensi, pilih Harian.
11. Untuk Format output, pilih CSV.

12. Untuk Status, pilih Aktifkan.
13. Di bagian enkripsi sisi server, pilih Nonaktifkan untuk tutorial ini.

    Untuk informasi selengkapnya, lihat Mengonfigurasi inventaris dengan menggunakan konsol S3 dan Memberikan izin kepada Amazon S3 untuk menggunakan CMK untuk enkripsi.
14. Di bidang tambahan - bagian opsional, pilih Ukuran, Terakhir dimodifikasi, dan kelas Penyimpanan.
15. Pilih Buat.

Untuk informasi selengkapnya, lihat Mengonfigurasi inventaris dengan menggunakan konsol S3.

Periksa laporan inventaris untuk bucket sumber video S3 Anda

Saat laporan inventaris diterbitkan, file manifes dikirim ke bucket tujuan Inventaris S3.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dalam daftar Bucket, pilih nama bucket sumber video (misalnya,**`tutorial-bucket-1`**).
4. Pilih Manajemen.
5. Untuk melihat apakah laporan Inventaris S3 Anda sudah siap sehingga Anda dapat membuat pekerjaan Operasi Batch S3 di Langkah 7, di bawah konfigurasi Inventaris, periksa apakah tombol Buat pekerjaan dari manifes diaktifkan.

   > **Note**
   >
   > Diperlukan waktu hingga 48 jam untuk mengirimkan laporan inventaris pertama. Jika tombol Buat pekerjaan dari manifes dinonaktifkan, laporan inventaris pertama belum dikirim. Tunggu hingga laporan inventaris pertama dikirimkan dan tombol Buat pekerjaan dari manifes diaktifkan sebelum Anda membuat pekerjaan Operasi Batch S3 di Langkah 7.
6. Untuk memeriksa laporan Inventaris S3 (`manifest.json`), di kolom Tujuan, pilih nama bucket tujuan inventaris yang Anda buat sebelumnya untuk menyimpan laporan inventaris (misalnya,**`tutorial-bucket-3`**).

7. Pada tab Objects, pilih folder yang ada dengan nama bucket sumber S3 Anda (misalnya,**`tutorial-bucket-1`**). Kemudian pilih nama yang Anda masukkan dalam nama konfigurasi Inventaris saat Anda membuat konfigurasi inventaris sebelumnya (misalnya,**`tutorial-inventory-config`**).

   Anda dapat melihat daftar folder dengan tanggal pembuatan laporan sebagai namanya.
8. Untuk memeriksa laporan Inventaris S3 harian untuk tanggal tertentu, pilih folder dengan nama tanggal pembuatan yang sesuai, lalu pilih`manifest.json`.
9. Untuk memeriksa detail laporan inventaris pada tanggal tertentu, pada halaman manifest.json, pilih Unduh atau Buka.

Langkah 6: Buat peran IAM untuk Operasi Batch S3

Untuk menggunakan Operasi Batch S3 untuk melakukan transcoding batch, Anda harus terlebih dahulu membuat peran IAM untuk memberikan izin Amazon S3 untuk melakukan Operasi Batch S3.

Sublangkah

- Membuat kebijakan IAM untuk Operasi Batch S3
- Buat peran IAM Operasi Batch S3 dan lampirkan kebijakan izin

Membuat kebijakan IAM untuk Operasi Batch S3

Anda harus membuat kebijakan IAM yang memberikan izin Operasi Batch S3 untuk membaca manifes masukan, menjalankan fungsi Lambda, dan menulis laporan penyelesaian pekerjaan Operasi Batch S3.

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di sebelah kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Pilih tab JSON.
5. Di bidang teks JSON, tempel kebijakan JSON berikut.

   Dalam kebijakan JSON, ganti empat nilai contoh dengan nilai berikut:

- Nama bucket sumber yang menyimpan video masukan Anda (misalnya,*tutorial-bucket-1*).
- Nama bucket tujuan inventaris yang Anda buat di Langkah 5 untuk menyimpan `manifest.json` file (misalnya,*tutorial-bucket-3*).
- Nama bucket yang Anda buat di Langkah 1 untuk menyimpan file media keluaran (misalnya,*tutorial-bucket-2*). Dalam tutorial ini, kita menempatkan laporan penyelesaian pekerjaan di bucket tujuan untuk file media output.
- Peran ARN dari fungsi Lambda yang Anda buat di Langkah 4. Untuk menemukan dan menyalin peran ARN dari fungsi Lambda, lakukan hal berikut:
  - Di tab browser baru, buka halaman Fungsi di konsol Lambda di. https://console.aws.amazon.com/lambda/home#/functions
  - Dalam daftar Fungsi, pilih nama fungsi Lambda yang Anda buat di Langkah 4 (misalnya,**`tutorial-lambda-convert`**).
  - Pilih Salin ARN.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3Get",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::tutorial-bucket-1/*",
                "arn:aws:s3:::tutorial-bucket-3/*"
            ]
        },
        {
            "Sid": "S3PutJobCompletionReport",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::tutorial-bucket-2/*"
        },
        {
            "Sid": "S3BatchOperationsInvokeLambda",
```

```
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": [
                "arn:aws:lambda:us-west-2:111122223333:function:tutorial-lambda-convert"
            ]
        }
    ]
}
```

6. Pilih Berikutnya: Tanda.
7. Pilih Berikutnya: Tinjau.
8. Di bidang Nama, masukkan **`tutorial-s3batch-policy`**.
9. Pilih Buat kebijakan.

Buat peran IAM Operasi Batch S3 dan lampirkan kebijakan izin

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi kiri, pilih Peran, lalu pilih Buat peran.
3. Pilih jenis Layanan AWSperan, lalu pilih layanan S3.
4. Di bawah Pilih kasus penggunaan Anda, pilih Operasi Batch S3.
5. Pilih Berikutnya: Izin.
6. Di bawah Lampirkan kebijakan izin, masukkan nama kebijakan IAM yang Anda buat sebelumnya (misalnya,**`tutorial-s3batch-policy`**) di kotak pencarian untuk memfilter daftar kebijakan. Pilih kotak centang di samping nama kebijakan (misalnya,**`tutorial-s3batch-policy`**).
7. Pilih Berikutnya: Tanda.
8. Pilih Selanjutnya: Tinjau.
9. Untuk Nama peran, masukkan **`tutorial-s3batch-role`**.
10. Pilih Buat peran.

    Setelah Anda membuat peran IAM untuk Operasi Batch S3, kebijakan kepercayaan berikut secara otomatis dilampirkan ke peran tersebut. Kebijakan kepercayaan ini memungkinkan prinsipal layanan Operasi Batch S3 untuk mengambil peran IAM.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"batchoperations.s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

Langkah 7: Buat dan jalankan pekerjaan Operasi Batch S3

Untuk membuat tugas Operasi Batch S3 untuk memproses video input di bucket sumber S3, Anda harus menentukan parameter untuk pekerjaan khusus ini.

> **Note**
>
> Sebelum Anda mulai membuat pekerjaan Operasi Batch S3, pastikan tombol Buat pekerjaan dari manifes diaktifkan. Untuk informasi selengkapnya, lihat Periksa laporan inventaris untuk bucket sumber video S3 Anda. Jika tombol Buat pekerjaan dari manifes dinonaktifkan, laporan inventaris pertama belum terkirim dan Anda harus menunggu hingga tombol diaktifkan. Setelah mengonfigurasi Inventaris Amazon S3 untuk bucket sumber S3 di Langkah 5, diperlukan waktu hingga 48 jam untuk mengirimkan laporan inventaris pertama.

Sublangkah

- Membuat tugas Operasi Batch S3
- Jalankan tugas Operasi Batch S3 untuk menginvokasi fungsi Lambda Anda
- (Opsional) Memeriksa laporan penyelesaian Anda
- (Opsional) Memantau setiap invokasi Lambda di konsol Lambda
- (Opsional) Pantau setiap pekerjaan MediaConvert transcoding video di konsol MediaConvert

Membuat tugas Operasi Batch S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Operasi Batch.
3. Pilih Buat tugas.
4. Untuk Wilayah AWS, pilih Wilayah tempat Anda ingin membuat tugas Anda.

   Dalam tutorial ini, untuk menggunakan tugas Operasi Batch S3 untuk menginvokasi fungsi Lambda, Anda harus membuat tugas di Wilayah yang sama dengan bucket sumber video S3 tempat objek yang direferensikan dalam manifes tersebut berada.
5. Di bagian Manifes, lakukan hal berikut ini:

   a. Untuk Format manifes, pilih Laporan Inventaris S3 (manifest.json).

   b. Untuk Objek manifes, pilih Telusuri S3 untuk menemukan bucket yang Anda buat di Langkah 5 untuk menyimpan laporan inventaris (misalnya, **`tutorial-bucket-3`**). Pada halaman Objek manifes, navigasikan nama objek hingga Anda menemukan file `manifest.json` untuk tanggal tertentu. File ini mencantumkan informasi tentang semua video yang transkode batch-nya ingin Anda lakukan. Ketika Anda telah menemukan file `manifest.json` yang ingin Anda gunakan, pilih tombol opsi di sebelahnya. Kemudian, pilih Pilih jalur.

   c. (Opsional) Untuk ID versi objek manifes-opsional, masukkan ID versi untuk objek manifes jika Anda ingin menggunakan versi selain yang terbaru.
6. Pilih Selanjutnya.
7. Untuk menggunakan fungsi Lambda untuk mentranskode semua objek yang terdaftar dalam file `manifest.json` yang dipilih, di bawah Jenis operasi, pilih Menginvokasi fungsi AWS Lambda .
8. Di bagian Menginvokasi fungsi Lambda, lakukan hal berikut ini:

   a. Pilih Pilih dari fungsi di akun Anda.

   b. Untuk Fungsi Lambda, pilih nama fungsi Lambda yang telah Anda buat di Langkah 4 (misalnya, **`tutorial-lambda-convert`**).

   c. Untuk Versi fungsi Lambda, pertahankan nilai default $LATEST.
9. Pilih Selanjutnya. Halaman Konfigurasi opsi tambahan terbuka.
10. Di bagian Opsi tambahan, pertahankan pengaturan default.

Untuk informasi selengkapnya tentang opsi ini, lihat Elemen permintaan pekerjaan Operasi Batch.

11. Di bagian Laporan penyelesaian, untuk Jalur ke tujuan laporan penyelesaian, pilih Telusuri S3. Temukan bucket yang telah Anda buat untuk file media output di Langkah 1 (misalnya, **`tutorial-bucket-2`**). Pilih tombol opsi di sebelah nama bucket tersebut. Kemudian, pilih Pilih jalur.

    Untuk pengaturan Laporan penyelesaian yang tersisa, pertahankan defaultnya. Untuk informasi selengkapnya tentang pengaturan laporan penyelesaian, lihat Elemen permintaan pekerjaan Operasi Batch. Laporan penyelesaian menyimpan catatan detail tugas dan operasi yang dilakukan.

12. Di bagian Izin, pilih Pilih dari peran IAM yang ada. Untuk Peran IAM, pilih peran IAM untuk tugas Operasi Batch S3 yang Anda buat di Langkah 6 (misalnya, **`tutorial-s3batch-role`**).
13. Pilih Selanjutnya.
14. Pada halaman Ulasan, tinjau pengaturannya. Lalu, pilih Buat tugas.

    Setelah S3 selesai membaca manifes tugas Operasi Batch S3 Anda, itu akan menetapkan Status tugas menjadi Menunggu konfirmasi Anda untuk menjalankan. Untuk melihat pembaruan status tugas, segarkan halaman. Anda tidak dapat menjalankan tugas Anda sampai statusnya Menunggu konfirmasi Anda untuk menjalankan.

Jalankan tugas Operasi Batch S3 untuk menginvokasi fungsi Lambda Anda

Jalankan tugas Operasi Batch Anda untuk menginvokasi fungsi Lambda untuk transkode video. Jika tugas Anda gagal, Anda dapat memeriksa laporan penyelesaian Anda untuk mengidentifikasi penyebabnya.

Untuk menjalankan tugas Operasi Batch S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Operasi Batch.
3. Dalam daftar Tugas, pilih ID tugas di baris pertama, yang merupakan tugas Operasi Batch S3 yang telah Anda buat sebelumnya.
4. Pilih Jalankan tugas.

5. Tinjau kembali parameter tugas Anda, dan konfirmasikan bahwa nilai untuk Total objek yang tercantum dalam temuan sama dengan jumlah objek dalam manifes tersebut. Lalu, pilih Jalankan tugas.

   Halaman tugas Operasi Batch S3 Anda terbuka.

6. Setelah tugas mulai berjalan, pada halaman tugas Anda, di bawah Status, periksa kemajuan tugas Operasi Batch S3 Anda, seperti Status, % Selesai, Total yang berhasil (laju), Total yang gagal (laju), Tanggal dihentikan, dan Alasan terminasi.

   Saat tugas Operasi Batch S3 selesai, lihat data di halaman tugas Anda untuk mengonfirmasi bahwa tugas selesai seperti yang diharapkan.

   Jika lebih dari 50 persen operasi objek tugas Operasi Batch S3 gagal setelah lebih dari 1.000 pengoperasian yang telah dicoba, tugas tersebut secara otomatis gagal. Untuk memeriksa laporan penyelesaian Anda guna mengidentifikasi penyebab kegagalan, gunakan prosedur opsional berikut ini.

(Opsional) Memeriksa laporan penyelesaian Anda

Anda dapat menggunakan laporan penyelesaian untuk menentukan objek mana yang gagal, serta penyebab kegagalannya.

Untuk memeriksa laporan penyelesaian Anda terkait detail mengenai objek yang gagal

1. Pada halaman tugas Operasi Batch S3 Anda, gulir ke bawah ke bagian Laporan penyelesaian, dan pilih tautan di bawah Tujuan laporan penyelesaian.

   Halaman bucket tujuan output S3 terbuka.

2. Pada tab Objek, pilih folder yang memiliki nama yang diakhiri dengan ID tugas Operasi Batch S3 yang telah Anda buat sebelumnya.
3. Pilih hasil/.
4. Centang kotak di samping file `.csv`.
5. Untuk melihat laporan tugas, pilih Buka atau Unduh.

(Opsional) Memantau setiap invokasi Lambda di konsol Lambda

Setelah tugas Operasi Batch S3 mulai berjalan, tugas akan memanggil fungsi Lambda untuk setiap objek video input. S3 menulis log dari setiap pemanggilan CloudWatch Lambda ke Log. Anda dapat menggunakan dasbor pemantauan konsol Lambda untuk memantau fungsi Lambda Anda.

1. Buka AWS Lambda konsol di https://console.aws.amazon.com/lambda/.
2. Di panel navigasi kiri, pilih Fungsi.
3. Dalam daftar Fungsi, pilih nama fungsi Lambda yang Anda buat di Langkah 4 (misalnya, **`tutorial-lambda-convert`**).
4. Pilih tab Monitor.
5. Di bawah Metrik, lihat metrik runtime untuk fungsi Lambda Anda.
6. Di bawah Log, lihat data log untuk setiap pemanggilan Lambda melalui Wawasan Log. CloudWatch

> **Note**
>
> Saat Anda menggunakan Operasi Batch S3 dengan fungsi Lambda, fungsi Lambda dipanggil pada setiap objek. Jika tugas Operasi Batch S3 Anda berukuran besar, tugas ini dapat menginvokasi beberapa fungsi Lambda secara bersamaan, yang menyebabkan lonjakan konkurensi Lambda.
> Masing-masing Akun AWS memiliki kuota konkurensi Lambda per Wilayah. Untuk informasi selengkapnya, lihat AWS Lambda Penskalaan Fungsi dalam AWS Lambda Panduan Pengembang. Praktik terbaik untuk menggunakan fungsi Lambda dengan Operasi Batch S3 adalah dengan menetapkan batas konkurensi pada fungsi Lambda itu sendiri. Menetapkan batas konkurensi membuat tugas tidak menghabiskan sebagian besar konkurensi Lambda Anda, dan berpotensi throttling fungsi lain di akun Anda. Untuk informasi selengkapnya, lihat Mengelola konkurensi cadangan Lambda di AWS Lambda Panduan Pengembang.

(Opsional) Pantau setiap pekerjaan MediaConvert transcoding video di konsol MediaConvert

MediaConvert Pekerjaan melakukan pekerjaan transcoding file media. Saat pekerjaan Operasi Batch S3 Anda memanggil fungsi Lambda Anda untuk setiap video, setiap pemanggilan fungsi Lambda membuat pekerjaan transcoding untuk setiap video input. MediaConvert

1. Masuk ke AWS Management Console dan buka MediaConvert konsol di https://console.aws.amazon.com/mediaconvert/.
2. Jika halaman MediaConvert pengantar muncul, pilih Memulai.
3. Dari daftar Tugas, lihat setiap baris untuk memantau tugas transkode bagi setiap video input.
4. Identifikasi baris tugas yang ingin Anda periksa, dan pilih tautan ID tugas untuk membuka halaman detail tugas.
5. Pada halaman Ringkasan tugas, di bagian bawah Output, pilih tautan untuk output HLS, MP4, atau Thumbnail, tergantung pada apa yang didukung oleh browser Anda, untuk membuka bucket tujuan S3 untuk file media output.
6. Di folder yang sesuai (HLS, MP4, atau Thumbnail) bucket tujuan output S3 Anda, pilih nama objek file media output.

   Halaman detail objek terbuka.
7. Pada halaman detail objek, di bawah Ikhtisar objek, pilih tautan di bawah URL objek untuk menonton file media output yang telah ditranskode.

Langkah 8: Memeriksa file media output dari bucket tujuan S3 Anda

Untuk memeriksa file media output dari bucket tujuan S3 Anda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dalam daftar Bucket, pilih nama bucket tujuan S3 untuk file media output yang Anda buat di Langkah 1 (misalnya, **`tutorial-bucket-2`**).
4. Pada tab Objek, setiap video input memiliki folder yang memiliki nama video input. Setiap folder berisi file media output transkode untuk video input.

   Untuk memeriksa file media output untuk video input, lakukan hal berikut ini:

a. Pilih folder dengan nama video input yang ingin Anda periksa.

b. Pilih folder Default/.

c. Pilih folder untuk format transkode (HLS, MP4, atau thumbnail dalam tutorial ini).

d. Pilih nama file media output.

e. Untuk melihat file yang ditranskode, pada halaman detail objek, pilih tautan di bawah URL Objek.

   File media output dalam format HLS dibagi menjadi segmen-segmen yang pendek. Untuk memutar video ini, sematkan URL objek file `.m3u8` di pemutar yang kompatibel.

Langkah 9: Membersihkan

Jika Anda melakukan transkode video menggunakan Operasi Batch S3, Lambda, MediaConvert dan hanya sebagai latihan pembelajaran, hapus AWS sumber daya yang Anda alokasikan sehingga Anda tidak lagi dikenakan biaya.

Sublangkah

- Hapus konfigurasi Inventaris S3 untuk bucket sumber S3 Anda
- Hapus fungsi Lambda
- Hapus grup CloudWatch log
- Menghapus peran IAM bersama dengan kebijakan inline untuk peran IAM
- Hapus kebijakan IAM yang dikelola pelanggan
- Kosongkan bucket S3
- Menghapus bucket S3

Hapus konfigurasi Inventaris S3 untuk bucket sumber S3 Anda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dalam daftar Bucket, pilih nama bucket sumber Anda (misalnya, **`tutorial-bucket-1`**).
4. Pilih tab Manajemen.

5. Di bagian Konfigurasi inventaris, pilih tombol opsi di sebelah konfigurasi inventaris yang Anda buat di Langkah 5 (misalnya, **`tutorial-inventory-config`**).
6. Pilih Hapus, lalu pilih Konfirmasi.

Hapus fungsi Lambda

1. Buka AWS Lambda konsol di https://console.aws.amazon.com/lambda/.
2. Di panel navigasi kiri, pilih Fungsi.
3. Centang kotak di samping fungsi yang Anda buat di Langkah 4 (misalnya, **`tutorial-lambda-convert`**).
4. Pilih Tindakan, lalu pilih Hapus.
5. Di kotak dialog Hapus fungsi, pilih Hapus.

Hapus grup CloudWatch log

1. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Di panel navigasi sebelah kiri, pilih Log, lalu pilih Grup log.
3. Pilih kotak centang di samping grup log yang memiliki nama yang diakhiri dengan fungsi Lambda yang Anda buat di Langkah 4 (misalnya, **`tutorial-lambda-convert`**).
4. Pilih Tindakan, lalu pilih Hapus grup log.
5. Di kotak dialog Hapus grup log, pilih Hapus.

Menghapus peran IAM bersama dengan kebijakan inline untuk peran IAM

Untuk menghapus peran IAM yang Anda buat di Langkah 2, Langkah 3, dan Langkah 6, lakukan hal berikut ini:

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran, lalu centang kotak di samping nama peran yang ingin Anda hapus.
3. Pilih Hapus di bagian atas halaman.

4. Di dalam kotak dialog konfirmasi, masukkan respons yang diperlukan di dalam kolom input teks berdasarkan prompt, lalu pilih Hapus.

Hapus kebijakan IAM yang dikelola pelanggan

Untuk menghapus kebijakan IAM yang dikelola pelanggan yang Anda buat di Langkah 6, lakukan hal berikut ini:

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di sebelah kiri, pilih Kebijakan.
3. Pilih tombol opsi di samping kebijakan yang Anda buat di Langkah 6 (misalnya, **`tutorial-s3batch-policy`**). Anda dapat menggunakan kotak pencarian untuk memfilter daftar kebijakan.
4. Pilih Tindakan, lalu pilih Hapus.
5. Konfirmasi bahwa Anda ingin menghapus kebijakan ini dengan memasukkan namanya di kolom teks, lalu pilih Hapus.

Kosongkan bucket S3

Untuk mengosongkan bucket S3 yang Anda buat di Prasyarat, Langkah 1, dan Langkah 5, lakukan hal berikut ini:

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di daftar Bucket, pilih tombol opsi di samping nama bucket yang ingin Anda kosongkan, lalu pilih Kosongkan.
4. Di halaman Bucket kosong, konfirmasikan bahwa Anda ingin mengosongkan bucket dengan mengetik **`permanently delete`** ke dalam bidang teks, lalu pilih Kosongkan.

Menghapus bucket S3

Untuk menghapus bucket S3 yang Anda buat di Prasyarat, Langkah 1, dan Langkah 5, lakukan hal berikut ini:

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di dalam daftar Bucket, pilih tombol opsi di samping nama bucket yang ingin Anda hapus.
4. Pilih Hapus.
5. Di halaman Hapus bucket, konfirmasikan bahwa Anda ingin menghapus bucket dengan memasukkan nama bucket ke dalam bidang teks, lalu pilih Hapus bucket.

Langkah selanjutnya

Setelah menyelesaikan tutorial ini, Anda dapat menjelajahi lebih lanjut kasus penggunaan relevan lainnya:

- Anda dapat menggunakan Amazon CloudFront untuk melakukan streaming file media yang ditranskode ke pemirsa di seluruh dunia. Untuk informasi selengkapnya, lihat Tutorial: Hosting video streaming sesuai permintaan dengan Amazon S3, Amazon, dan CloudFront Amazon Route 53.
- Anda dapat mentranskode video saat Anda mengunggahnya ke bucket sumber S3. Untuk melakukannya, Anda dapat mengonfigurasi pemicu peristiwa Amazon S3 yang secara otomatis memanggil fungsi Lambda untuk mentranskode objek baru di S3. MediaConvert Untuk informasi selengkapnya, lihat Tutorial: Menggunakan pemicu Amazon S3 untuk menginvokasi fungsi Lambda di AWS Lambda Panduan Pengembang.

Tutorial: Mengonfigurasi situs web statis untuk Amazon S3

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Anda dapat mengonfigurasi bucket Amazon S3 agar berfungsi seperti situs web. Contoh ini memandu Anda melalui langkah-langkah untuk menghosting situs web di Amazon S3.

> **Important**
>
> Tutorial berikut ini mengharuskan Blokir Akses Publik dinonaktifkan. Kami menyarankan agar Anda menyimpan Blokir Akses Publik diaktifkan. Jika Anda ingin mengaktifkan keempat pengaturan Blokir Akses Publik dan meng-host situs web statis, Anda dapat menggunakan Amazon CloudFront Origin Access Control (OAC). Amazon CloudFront menyediakan kemampuan yang diperlukan untuk menyiapkan situs web statis yang aman. Situs web statis Amazon S3 hanya mendukung titik akhir HTTP. Amazon CloudFront menggunakan penyimpanan Amazon S3 yang tahan lama sambil menyediakan header keamanan tambahan, seperti HTTPS. HTTPS menambahkan keamanan dengan mengenkripsi permintaan HTTP normal, dan melindungi dari serangan siber umum. Untuk informasi selengkapnya, lihat Memulai situs web statis aman di Panduan CloudFront Pengembang Amazon.

Topik

- Langkah 1: Buat bucket
- Langkah 2: Mengaktifkan hosting situs web statis
- Langkah 3: Mengedit pengaturan Blokir Akses Publik
- Langkah 4: Menambahkan kebijakan bucket yang membuat konten bucket Anda tersedia untuk umum
- Langkah 5: Mengonfigurasi dokumen indeks
- Langkah 6: Mengonfigurasi dokumen kesalahan
- Langkah 7: Menguji titik akhir situs web Anda
- Langkah 8: Membersihkan

Langkah 1: Buat bucket

Petunjuk berikut ini memberikan ikhtisar tentang cara membuat bucket Anda untuk menghosting situs web. Untuk detail, step-by-step instruksi tentang membuat ember, lihatMembuat bucket.

Untuk membuat bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pilih Buat bucket.
3. Masukkan Nama bucket (misalnya, **`example.com`**).
4. Pilih Wilayah tempat Anda ingin membuat bucket.

   Pilih Wilayah yang secara geografis dekat dengan Anda, untuk meminimalkan latensi serta biaya, atau untuk memenuhi persyaratan peraturan. Wilayah yang Anda pilih menentukan titik akhir situs web Amazon S3 Anda. Untuk informasi selengkapnya, lihat Titik akhir situs web.
5. Untuk menerima pengaturan default dan membuat bucket, pilih Buat.

Langkah 2: Mengaktifkan hosting situs web statis

Setelah membuat bucket, Anda dapat mengaktifkan hosting situs web statis untuk bucket Anda. Anda dapat membuat bucket baru, atau menggunakan bucket yang sudah ada.

Untuk mengaktifkan hosting situs web statis

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang ingin Anda aktifkan hosting situs web statisnya.
3. Pilih Properti.
4. Di bagian bawah Hosting situs web statis, pilih Edit.
5. Pilih Gunakan bucket ini untuk menghosting situs web.
6. Di bagian bawah Hosting situs web statis, pilih Aktifkan.
7. Di Dokumen indeks, masukkan nama file dokumen indeks, biasanya `index.html`.

   Nama dokumen indeks peka huruf besar/kecil, dan harus sama persis dengan nama file dokumen indeks HTML yang ingin Anda unggah ke bucket S3 Anda. Saat Anda mengonfigurasi bucket untuk hosting situs web, Anda harus menentukan dokumen indeks. Amazon S3 mengembalikan dokumen indeks ini ketika permintaan dibuat ke domain root atau subfolder mana pun. Untuk informasi selengkapnya, lihat Mengonfigurasi dokumen indeks.
8. Agar dapat menyediakan dokumen kesalahan kustom Anda sendiri untuk kesalahan kelas 4XX, di Dokumen kesalahan, masukkan nama file dokumen kesalahan kustom.

Nama dokumen kesalahan peka huruf besar/kecil, dan harus sama persis dengan nama file dokumen indeks HTML yang ingin Anda unggah ke bucket S3 Anda. Jika Anda tidak menentukan dokumen kesalahan khusus dan terjadi kesalahan, Amazon S3 mengembalikan dokumen kesalahan HTML default. Untuk informasi selengkapnya, lihat Mengonfigurasi dokumen kesalahan khusus.

9. (Opsional) Jika Anda ingin menentukan aturan pengalihan lanjutan, dalam Aturan pengalihan, masukkan JSON untuk menjelaskan aturannya.

   Misalnya, Anda dapat merutekan permintaan secara kondisional dengan nama kunci atau prefiks objek tertentu dalam permintaan tersebut. Untuk informasi selengkapnya, lihat Konfigurasikan aturan pengalihan untuk menggunakan pengalihan bersyarat lanjutan.

10. Pilih Simpan perubahan.

    Amazon S3 memungkinkan hosting situs web statis untuk bucket Anda. Di bagian bawah halaman, di bawah Hosting situs web statis, Anda melihat titik akhir situs web untuk bucket Anda.

11. Di bagian bawah Hosting situs web statis, perhatikan Titik Akhir.

    Titik Akhir adalah titik akhir situs web Amazon S3 untuk bucket Anda. Setelah Anda menyelesaikan konfigurasi bucket Anda sebagai situs web statis, Anda dapat menggunakan titik akhir ini untuk menguji situs web Anda.

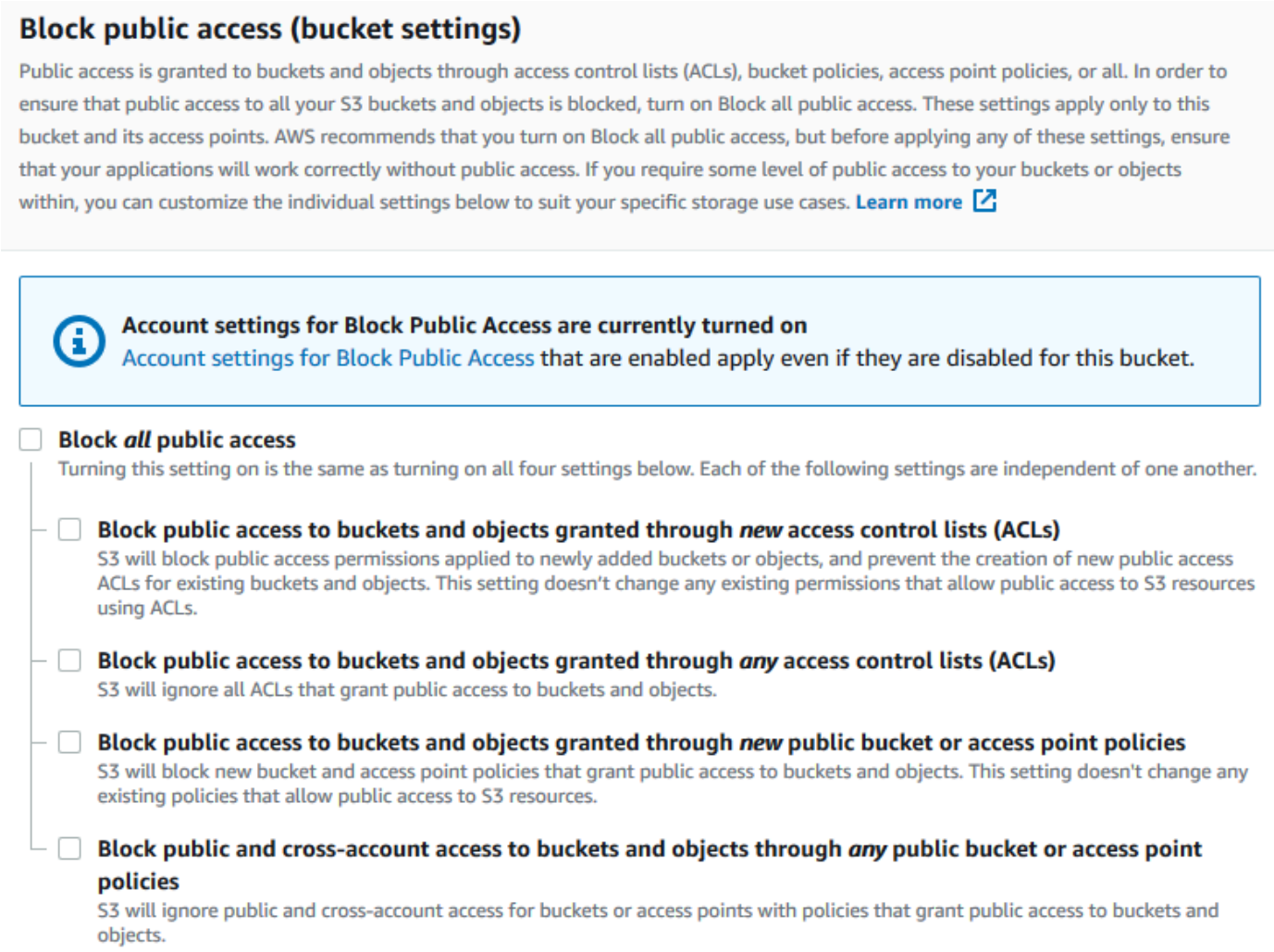
Langkah 3: Mengedit pengaturan Blokir Akses Publik

Secara default, Amazon S3 memblokir akses publik ke akun dan bucket Anda. Jika Anda ingin menggunakan bucket untuk menghosting situs web statis, Anda dapat menggunakan langkah-langkah ini untuk mengedit pengaturan blokir akses publik Anda.

> **Warning**
>
> Sebelum Anda menyelesaikan langkah ini, tinjau Melakukan blok akses publik ke penyimpanan Amazon S3 Anda untuk memastikan bahwa Anda telah memahami dan menerima risiko yang terkait dengan mengizinkan akses publik. Saat Anda mematikan pengaturan blokir akses publik untuk membuat bucket Anda menjadi publik, siapa pun di internet dapat mengakses bucket Anda. Kami sarankan agar Anda memblokir semua akses publik ke bucket Anda.

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pilih nama bucket yang telah Anda konfigurasikan sebagai situs web statis.
3. Pilih Izin.
4. Di bagian bawah Blokir akses publik (pengaturan bucket), pilih Edit.
5. Kosongkan Blokir semua akses publik, lalu pilih Simpan perubahan.

> **Warning**
>
> Sebelum Anda menyelesaikan langkah ini, tinjau Melakukan blok akses publik ke penyimpanan Amazon S3 Anda untuk memastikan bahwa Anda telah memahami dan menerima risiko yang terkait dengan mengizinkan akses publik. Saat Anda mematikan pengaturan blokir akses publik untuk membuat bucket Anda menjadi publik, siapa pun di internet dapat mengakses bucket Anda. Kami sarankan agar Anda memblokir semua akses publik ke bucket Anda.



Amazon S3 menonaktifkan pengaturan Blokir Akses Publik untuk bucket Anda. Untuk membuat situs web publik statis, Anda mungkin harus mengedit pengaturan Blokir Akses Publik untuk akun Anda sebelum menambahkan kebijakan bucket. Jika pengaturan akun untuk Blokir Akses Publik saat ini diaktifkan, Anda akan melihat catatan di Blokir akses publik (pengaturan bucket).

Langkah 4: Menambahkan kebijakan bucket yang membuat konten bucket Anda tersedia untuk umum

Setelah Anda mengedit pengaturan Blokir Akses Publik S3, Anda dapat menambahkan kebijakan bucket untuk memberikan akses baca publik ke bucket Anda. Saat Anda memberikan akses baca publik, siapa pun di internet dapat mengakses bucket Anda.

> **Important**
>
> Kebijakan berikut ini hanya merupakan contoh, dan memungkinkan akses penuh ke konten bucket Anda. Sebelum melanjutkan langkah ini, tinjau Bagaimana saya dapat mengamankan file dalam bucket Amazon S3 saya? untuk memastikan bahwa Anda telah memahami praktik terbaik untuk mengamankan file dalam bucket S3, dan risiko yang terlibat dalam pemberian akses publik.

1. Di bagian bawah Bucket, pilih nama bucket Anda.
2. Pilih Izin.
3. Di Bawah Kebijakan bucket, pilih Edit.
4. Untuk memberikan akses baca bagi publik untuk situs web Anda, salin kebijakan kelompok berikut, dan tempelkan di Editor kebijakan bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::Bucket-Name/*"
            ]
        }
    ]
}
```

5. Perbarui `Resource` dengan nama bucket Anda.

   Dalam contoh kebijakan bucket sebelumnya, *`Nama-Bucket`* adalah placeholder untuk nama bucket tersebut. Untuk menggunakan kebijakan bucket ini dengan bucket Anda sendiri, Anda harus memperbarui nama ini agar sesuai dengan nama bucket Anda.
6. Pilih Simpan perubahan.

Pesan akan muncul, yang menunjukkan bahwa kebijakan bucket telah berhasil ditambahkan.

Jika Anda melihat kesalahan yang mengatakan `Policy has invalid resource`, konfirmasikan bahwa nama bucket dalam kebijakan bucket tersebut sesuai dengan nama bucket Anda. Untuk informasi tentang menambahkan kebijakan bucket, lihat Bagaimana cara menambahkan kebijakan S3 bucket?

Jika Anda mendapatkan pesan kesalahan dan tidak dapat menyimpan kebijakan bucket, periksa pengaturan akun dan bucket Blokir Akses Publik untuk mengonfirmasi bahwa Anda mengizinkan akses publik ke bucket.

Langkah 5: Mengonfigurasi dokumen indeks

Saat mengaktifkan hosting situs web statis untuk bucket Anda, Anda memasukkan nama dokumen indeks (misalnya, **`index.html`**). Setelah Anda mengaktifkan hosting situs web statis untuk bucket, Anda mengunggah file HTML dengan nama dokumen indeks ke bucket Anda.

Untuk mengonfigurasi dokumen indeks

1. Buat file `index.html`.

   Jika Anda tidak memiliki file `index.html`, Anda dapat menggunakan HTML berikut ini untuk membuatnya:

   ```
   <html xmlns="http://www.w3.org/1999/xhtml" >
   <head>
       <title>My Website Home Page</title>
   </head>
   <body>
     <h1>Welcome to my website</h1>
     <p>Now hosted on Amazon S3!</p>
   </body>
   </html>
   ```

2. Simpan file indeks secara lokal.

   Nama file dokumen indeks harus sama persis dengan nama dokumen indeks yang Anda masukkan ke dalam kotak dialog Hosting situs web statis. Nama dokumen indeks peka huruf besar/kecil. Misalnya, jika Anda memasukkan `index.html` untuk Dokumen indeks dalam

kotak dialog Hosting situs web statis, nama file dokumen indeks Anda juga harus dan bukan `Index.html`.

3. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
4. Di daftar Bucket, pilih nama bucket yang ingin Anda gunakan untuk meng-host situs web statis.
5. Aktifkan hosting situs web statis untuk bucket Anda, lalu masukkan nama persis dokumen indeks Anda (misalnya, `index.html`). Untuk informasi selengkapnya, lihat Mengaktifkan hosting situs web.

   Setelah mengaktifkan hosting situs web statis, lanjutkan ke langkah 6.
6. Untuk mengunggah dokumen indeks ke bucket Anda, lakukan salah satu hal berikut ini:

   - Seret dan jatuhkan file indeks ke dalam daftar bucket konsol.
   - Pilih Unggah, dan ikuti petunjuk untuk memilih dan mengunggah file indeks.

   Untuk step-by-step instruksi, lihatMengunggah Objek.
7. (Opsional) Mengunggah konten situs web lain ke bucket Anda.

Langkah 6: Mengonfigurasi dokumen kesalahan

Saat mengaktifkan hosting situs web statis untuk bucket Anda, Anda memasukkan nama dokumen kesalahan (misalnya, **`404.html`**). Setelah Anda mengaktifkan hosting situs web statis untuk bucket, Anda mengunggah file HTML dengan nama dokumen indeks ke bucket Anda.

Untuk mengonfigurasi dokumen kesalahan

1. Membuat dokumen kesalahan, misalnya `404.html`.
2. Simpan file dokumen kesalahan secara lokal.

   Nama dokumen kesalahan peka huruf besar/kecil, dan harus sama persis dengan nama yang Anda masukkan saat Anda mengaktifkan hosting situs web statis. Misalnya, jika Anda memasukkan `404.html` sebagai nama Dokumen kesalahan di kotak dialog Hosting situs web statis, nama file dokumen kesalahan Anda juga harus bernama `404.html`.
3. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
4. Di daftar Bucket, pilih nama bucket yang ingin Anda gunakan untuk meng-host situs web statis.

5. Aktifkan hosting situs web statis untuk bucket Anda, lalu masukkan nama persis dokumen indeks Anda (misalnya, `404.html`). Untuk informasi selengkapnya, lihat Mengaktifkan hosting situs web dan Mengonfigurasi dokumen kesalahan khusus.

   Setelah mengaktifkan hosting situs web statis, lanjutkan ke langkah 6.

6. Untuk mengunggah dokumen kesalahan ke bucket Anda, lakukan salah satu hal berikut ini:

   - Seret dan jatuhkan file dokumen kesalahan ke dalam daftar bucket konsol.
   - Pilih Unggah, dan ikuti petunjuk untuk memilih dan mengunggah file indeks.

   Untuk step-by-step instruksi, lihatMengunggah Objek.

Langkah 7: Menguji titik akhir situs web Anda

Setelah Anda mengonfigurasi hosting situs web statis untuk bucket, Anda dapat menguji titik akhir situs web Anda.

> **Note**
>
> Amazon S3 tidak mendukung akses HTTPS ke situs web. Jika Anda ingin menggunakan HTTPS, Anda dapat menggunakan Amazon CloudFront untuk melayani situs web statis yang dihosting di Amazon S3.
> Untuk informasi selengkapnya, lihat Bagaimana CloudFront cara menggunakan situs web statis yang dihosting di Amazon S3? dan Memerlukan HTTPS untuk komunikasi antara pemirsa dan CloudFront.

1. Di bagian bawah Bucket, pilih nama bucket Anda.
2. Pilih Properti.
3. Di bagian bawah halaman, di bawah Hosting situs web statis, pilih Titik akhir situs web bucket.

   Dokumen indeks Anda terbuka di jendela browser terpisah.

Anda sekarang memiliki situs web yang di-host di Amazon S3. Situs web ini tersedia di titik akhir situs web Amazon S3. Namun, Anda mungkin memiliki domain, seperti `example.com`, yang ingin Anda gunakan untuk menyajikan konten dari situs web yang Anda buat. Anda mungkin juga ingin

menggunakan dukungan domain root Amazon S3 untuk melayani permintaan untuk keduanya `http://www.example.com` dan `http://example.com`. Ini memerlukan langkah-langkah tambahan. Sebagai contoh, lihat Tutorial: Mengonfigurasi situs web statis menggunakan domain kustom yang terdaftar di Route 53.

Langkah 8: Membersihkan

Jika Anda membuat situs web statis hanya sebagai latihan pembelajaran, hapus sumber daya AWS yang Anda alokasikan, sehingga Anda tidak lagi dikenakan biaya. Setelah Anda menghapus AWS sumber daya Anda, situs web Anda tidak lagi tersedia. Untuk informasi selengkapnya, lihat Menghapus bucket.

Tutorial: Mengonfigurasi situs web statis menggunakan domain kustom yang terdaftar di Route 53

Misalnya, Anda ingin menghosting situs web statis di Amazon S3. Anda telah mendaftarkan domain dengan Amazon Route 53 (misalnya, `example.com`), dan Anda ingin meminta `http://www.example.com` dan `http://example.com` untuk dilayani dari konten Amazon S3. Anda dapat menggunakan panduan ini untuk mempelajari cara menghosting situs web statis, dan membuat pengalihan ke Amazon S3 untuk situs web dengan nama domain kustom yang terdaftar di Route 53. Anda dapat bekerja dengan situs web yang ada yang ingin Anda hosting di Amazon S3, atau menggunakan panduan ini untuk memulai dari awal.

Setelah Anda menyelesaikan panduan ini, Anda dapat menggunakan Amazon secara opsional CloudFront untuk meningkatkan kinerja situs web Anda. Untuk informasi selengkapnya, lihat Mempercepat situs web Anda dengan Amazon CloudFront.

> **Note**
>
> Titik akhir situs web Amazon S3 tidak mendukung HTTPS atau titik akses. Jika Anda ingin menggunakan HTTPS, Anda dapat menggunakan Amazon CloudFront untuk melayani situs web statis yang dihosting di Amazon S3.
> Untuk tutorial tentang cara meng-host konten Anda dengan aman CloudFront dan Amazon S3, lihat. Tutorial: Hosting video streaming sesuai permintaan dengan Amazon S3, Amazon, dan CloudFront Amazon Route 53 Untuk informasi selengkapnya, lihat Bagaimana CloudFront cara menggunakan situs web statis yang dihosting di Amazon S3? dan Memerlukan HTTPS untuk komunikasi antara pemirsa dan CloudFront.

Mengotomatiskan pengaturan situs web statis dengan templat AWS CloudFormation

Anda dapat menggunakan AWS CloudFormation template untuk mengotomatiskan pengaturan situs web statis Anda. AWS CloudFormation Template mengatur komponen yang Anda butuhkan untuk meng-host situs web statis yang aman sehingga Anda dapat lebih fokus pada konten situs web Anda dan lebih sedikit pada konfigurasi komponen.

AWS CloudFormation Template mencakup komponen-komponen berikut:

- Amazon S3–Membuat bucket Amazon S3 untuk menghosting situs web statis Anda.
- CloudFront — Membuat CloudFront distribusi untuk mempercepat situs web statis Anda.
- Lambda@Edge–Menggunakan Lambda@Edge untuk menambahkan header keamanan ke setiap respons server. Header keamanan adalah sekelompok header di respons server web yang memberi tahu browser web untuk mengambil tindakan pencegahan keamanan ekstra. Untuk informasi selengkapnya, lihat posting blog Menambahkan header keamanan HTTP menggunakan Lambda @Edge dan Amazon. CloudFront

AWS CloudFormation Template ini tersedia untuk Anda unduh dan gunakan. Untuk informasi dan petunjuk, lihat Memulai situs web statis aman di Panduan CloudFront Pengembang Amazon.

Topik

- Sebelum Anda mulai
- Langkah 1: Mendaftarkan domain kustom dengan Route 53
- Langkah 2: Membuat dua bucket
- Langkah 3: Mengonfigurasi bucket domain root Anda untuk hosting situs web
- Langkah 4: Mengonfigurasi bucket subdomain Anda untuk pengalihan situs web
- Langkah 5: Mengonfigurasi pencatatan untuk lalu lintas situs web
- Langkah 6: Mengunggah konten indeks dan situs web
- Langkah 7: Mengunggah dokumen kesalahan
- Langkah 8: Edit pengaturan Blokir Akses Publik S3
- Langkah 9: Melampirkan kebijakan bucket
- Langkah 10: Menguji titik akhir domain Anda
- Langkah 11: Menambahkan catatan alias untuk domain dan subdomain Anda
- Langkah 12: Menguji situs webnya
- Mempercepat situs web Anda dengan Amazon CloudFront

- Membersihkan sumber daya contoh Anda

Sebelum Anda mulai

Saat Anda mengikuti langkah-langkah dalam contoh ini, Anda bekerja dengan layanan berikut:

Amazon Route 53–Anda menggunakan Route 53 untuk mendaftarkan domain dan menentukan di mana Anda ingin mengirimkan lalu lintas internet untuk domain Anda. Contoh ini menunjukkan cara untuk membuat catatan alias Route 53 yang merutekan lalu lintas untuk domain Anda (`example.com`) dan subdomain (`www.example.com`) ke bucket Amazon S3 yang berisi file HTML.

Amazon S3–Anda menggunakan Amazon S3 untuk membuat bucket, mengunggah halaman situs web sampel, mengonfigurasi izin sehingga semua orang dapat melihat konten, dan kemudian mengonfigurasi bucket untuk hosting situs web.

Langkah 1: Mendaftarkan domain kustom dengan Route 53

Jika Anda belum memiliki nama domain yang terdaftar, seperti `example.com`, daftarkan satu dengan Route 53. Untuk informasi selengkapnya, lihat Mendaftarkan nama domain di Panduan Pengembang Amazon Route 53. Setelah mendaftarkan nama domain Anda, Anda dapat membuat dan mengonfigurasi bucket Amazon S3 Anda untuk hosting situs web.

Langkah 2: Membuat dua bucket

Untuk mendukung permintaan dari domain root dan subdomain, Anda membuat dua bucket.

- Bucket domain–`example.com`
- Bucket subdomain–`www.example.com`

Nama bucket ini harus sama persis dengan nama domain Anda. Dalam contoh ini, nama domain adalah `example.com`. Anda menghosting konten Anda dari bucket domain root (`example.com`). Anda membuat permintaan pengalihan untuk bucket subdomain (`www.example.com`). Jika seseorang masuk ke `www.example.com` dalam browser mereka, mereka akan diarahkan ke `example.com` dan melihat konten yang di-host di bucket Amazon S3 dengan nama itu.

Untuk membuat bucket Anda untuk hosting situs web

Petunjuk berikut ini memberikan ikhtisar tentang cara membuat bucket Anda untuk menghosting situs web. Untuk detail, step-by-step instruksi tentang membuat ember, lihatMembuat bucket.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Buat bucket domain root Anda:
   a. Pilih Buat bucket.
   b. Masukkan Nama bucket (misalnya, **`example.com`**).
   c. Pilih Wilayah tempat Anda ingin membuat bucket.

      Pilih Wilayah yang secara geografis dekat dengan Anda, untuk meminimalkan latensi serta biaya, atau untuk memenuhi persyaratan peraturan. Wilayah yang Anda pilih menentukan titik akhir situs web Amazon S3 Anda. Untuk informasi selengkapnya, lihat Titik akhir situs web.
   d. Untuk menerima pengaturan default dan membuat bucket, pilih Buat.
3. Buat bucket subdomain Anda:
   a. Pilih Buat bucket.
   b. Masukkan Nama bucket (misalnya, **`www.example.com`**).
   c. Pilih Wilayah tempat Anda ingin membuat bucket.

      Pilih Wilayah yang secara geografis dekat dengan Anda, untuk meminimalkan latensi serta biaya, atau untuk memenuhi persyaratan peraturan. Wilayah yang Anda pilih menentukan titik akhir situs web Amazon S3 Anda. Untuk informasi selengkapnya, lihat Titik akhir situs web.
   d. Untuk menerima pengaturan default dan membuat bucket, pilih Buat.

Pada langkah berikutnya, Anda mengonfigurasi `example.com` untuk hosting situs web.

Langkah 3: Mengonfigurasi bucket domain root Anda untuk hosting situs web

Pada langkah ini, Anda mengonfigurasi bucket domain root (`example.com`) sebagai situs web. Bucket ini akan berisi konten situs web Anda. Saat Anda mengonfigurasikan bucket untuk hosting situs web, Anda dapat mengakses situs web menggunakan Titik akhir situs web.

Untuk mengaktifkan hosting situs web statis

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang ingin Anda aktifkan hosting situs web statisnya.
3. Pilih Properti.
4. Di bagian bawah Hosting situs web statis, pilih Edit.
5. Pilih Gunakan bucket ini untuk menghosting situs web.
6. Di bagian bawah Hosting situs web statis, pilih Aktifkan.
7. Di Dokumen indeks, masukkan nama file dokumen indeks, biasanya `index.html`.

   Nama dokumen indeks peka huruf besar/kecil, dan harus sama persis dengan nama file dokumen indeks HTML yang ingin Anda unggah ke bucket S3 Anda. Saat Anda mengonfigurasi bucket untuk hosting situs web, Anda harus menentukan dokumen indeks. Amazon S3 mengembalikan dokumen indeks ini ketika permintaan dibuat ke domain root atau subfolder mana pun. Untuk informasi selengkapnya, lihat Mengonfigurasi dokumen indeks.
8. Agar dapat menyediakan dokumen kesalahan kustom Anda sendiri untuk kesalahan kelas 4XX, di Dokumen kesalahan, masukkan nama file dokumen kesalahan kustom.

   Nama dokumen kesalahan peka huruf besar/kecil, dan harus sama persis dengan nama file dokumen indeks HTML yang ingin Anda unggah ke bucket S3 Anda. Jika Anda tidak menentukan dokumen kesalahan khusus dan terjadi kesalahan, Amazon S3 mengembalikan dokumen kesalahan HTML default. Untuk informasi selengkapnya, lihat Mengonfigurasi dokumen kesalahan khusus.
9. (Opsional) Jika Anda ingin menentukan aturan pengalihan lanjutan, dalam Aturan pengalihan, masukkan JSON untuk menjelaskan aturannya.

   Misalnya, Anda dapat merutekan permintaan secara kondisional dengan nama kunci atau prefiks objek tertentu dalam permintaan tersebut. Untuk informasi selengkapnya, lihat Konfigurasikan aturan pengalihan untuk menggunakan pengalihan bersyarat lanjutan.
10. Pilih Simpan perubahan.

    Amazon S3 memungkinkan hosting situs web statis untuk bucket Anda. Di bagian bawah halaman, di bawah Hosting situs web statis, Anda melihat titik akhir situs web untuk bucket Anda.
11. Di bagian bawah Hosting situs web statis, perhatikan Titik Akhir.

Titik Akhir adalah titik akhir situs web Amazon S3 untuk bucket Anda. Setelah Anda menyelesaikan konfigurasi bucket Anda sebagai situs web statis, Anda dapat menggunakan titik akhir ini untuk menguji situs web Anda.

Setelah Anda mengedit pengaturan blokir akses publik dan menambahkan kebijakan bucket yang memungkinkan akses baca publik, Anda dapat menggunakan titik akhir situs web untuk mengakses situs web Anda.

Pada langkah berikutnya, Anda mengonfigurasi subdomain (`www.example.com`) untuk mengalihkan permintaan ke domain Anda (`example.com`).

Langkah 4: Mengonfigurasi bucket subdomain Anda untuk pengalihan situs web

Setelah Anda mengonfigurasi bucket domain root untuk hosting situs web, Anda dapat mengonfigurasikan bucket subdomain untuk mengalihkan semua permintaan ke domain. Dalam contoh ini, semua permintaan untuk `www.example.com` diarahkan kembali ke `example.com`.

Untuk mengonfigurasi permintaan pengalihan

1. Di konsol Amazon S3, di Bucket Anda, pilih nama bucket subdomain Anda (`www.example.com` dalam contoh ini).
2. Pilih Properti.
3. Di bagian bawah Hosting situs web statis, pilih Edit.
4. Pilih Alihkan permintaan objek.
5. Di Bucket target, masukkan domain root Anda, misalnya, **`example.com`**.
6. Untuk Protokol, pilih http.
7. Pilih Simpan perubahan.

Langkah 5: Mengonfigurasi pencatatan untuk lalu lintas situs web

Jika ingin melacak jumlah pengunjung yang mengakses situs web Anda, Anda dapat mengaktifkan log untuk bucket domain root Anda secara opsional. Untuk informasi selengkapnya, lihat Pencatatan permintaan dengan pencatatan akses server. Jika Anda berencana menggunakan Amazon CloudFront untuk mempercepat situs web Anda, Anda juga dapat menggunakan CloudFront logging.

Untuk mengaktifkan log akses server untuk bucket domain root Anda

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Dalam Wilayah yang sama tempat Anda membuat bucket yang dikonfigurasi sebagai situs web statis, buat bucket untuk log, misalnya `logs.example.com`.
3. Buat folder untuk file log pencatatan akses server (misalnya, `logs`).
4. (Opsional) Jika Anda ingin menggunakan CloudFront untuk meningkatkan kinerja situs web Anda, buat folder untuk file CloudFront log (misalnya,`cdn`).

   > **Important**
   >
   > Saat Anda membuat atau memperbarui distribusi dan mengaktifkan CloudFront pencatatan, CloudFront perbarui daftar kontrol akses bucket (ACL) untuk memberikan `FULL_CONTROL` izin `awslogsdelivery` akun untuk menulis log ke bucket Anda. Untuk informasi selengkapnya, lihat Izin yang diperlukan untuk mengonfigurasi pencatatan standar dan mengakses file log Anda di Panduan CloudFront Pengembang Amazon. Jika bucket yang menyimpan log menggunakan setelan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek S3 untuk menonaktifkan ACL, CloudFront tidak dapat menulis log ke bucket. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

5. Di Bucket Anda, pilih bucket domain root Anda.
6. Pilih Properti.
7. Di bagian bawah Pencatatan akses server, pilih Edit.
8. Pilih Aktifkan.
9. Under the Target bucket, choose the bucket and folder destination for the server access:
   - Jelajahi ke lokasi folder dan bucket:
     1. Pilih Jelajahi S3.
     2. Pilih nama bucket, lalu pilih folder log.
     3. Pilih Pilih jalur.
   - Masukkan alur bucket S3, misalnya, `s3://logs.example.com/logs/`.
10. Pilih Simpan perubahan.

Dalam bucket log, sekarang dapat mengakses log Anda. Amazon S3 menulis log akses situs web ke bucket log Anda setiap 2 jam.

Langkah 6: Mengunggah konten indeks dan situs web

Dalam langkah ini, Anda mengunggah dokumen indeks dan konten situs web opsional ke bucket domain root Anda.

Saat mengaktifkan hosting situs web statis untuk bucket Anda, Anda memasukkan nama dokumen indeks (misalnya, **`index.html`**). Setelah Anda mengaktifkan hosting situs web statis untuk bucket, Anda mengunggah file HTML dengan nama dokumen indeks ke bucket Anda.

Untuk mengonfigurasi dokumen indeks

1. Buat file `index.html`.

   Jika Anda tidak memiliki file `index.html`, Anda dapat menggunakan HTML berikut ini untuk membuatnya:

   ```
   <html xmlns="http://www.w3.org/1999/xhtml" >
   <head>
       <title>My Website Home Page</title>
   </head>
   <body>
     <h1>Welcome to my website</h1>
     <p>Now hosted on Amazon S3!</p>
   </body>
   </html>
   ```

2. Simpan file indeks secara lokal.

   Nama file dokumen indeks harus sama persis dengan nama dokumen indeks yang Anda masukkan ke dalam kotak dialog Hosting situs web statis. Nama dokumen indeks peka huruf besar/kecil. Misalnya, jika Anda memasukkan `index.html` untuk Dokumen indeks dalam kotak dialog Hosting situs web statis, nama file dokumen indeks Anda juga harus dan bukan `Index.html`.

3. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
4. Di daftar Bucket, pilih nama bucket yang ingin Anda gunakan untuk meng-host situs web statis.

5. Aktifkan hosting situs web statis untuk bucket Anda, lalu masukkan nama persis dokumen indeks Anda (misalnya, `index.html`). Untuk informasi selengkapnya, lihat Mengaktifkan hosting situs web.

   Setelah mengaktifkan hosting situs web statis, lanjutkan ke langkah 6.

6. Untuk mengunggah dokumen indeks ke bucket Anda, lakukan salah satu hal berikut ini:

   - Seret dan jatuhkan file indeks ke dalam daftar bucket konsol.
   - Pilih Unggah, dan ikuti petunjuk untuk memilih dan mengunggah file indeks.

   Untuk step-by-step instruksi, lihatMengunggah Objek.

7. (Opsional) Mengunggah konten situs web lain ke bucket Anda.

Langkah 7: Mengunggah dokumen kesalahan

Saat mengaktifkan hosting situs web statis untuk bucket Anda, Anda memasukkan nama dokumen kesalahan (misalnya, **`404.html`**). Setelah Anda mengaktifkan hosting situs web statis untuk bucket, Anda mengunggah file HTML dengan nama dokumen indeks ke bucket Anda.

Untuk mengonfigurasi dokumen kesalahan

1. Membuat dokumen kesalahan, misalnya `404.html`.
2. Simpan file dokumen kesalahan secara lokal.

   Nama dokumen kesalahan peka huruf besar/kecil, dan harus sama persis dengan nama yang Anda masukkan saat Anda mengaktifkan hosting situs web statis. Misalnya, jika Anda memasukkan `404.html` sebagai nama Dokumen kesalahan di kotak dialog Hosting situs web statis, nama file dokumen kesalahan Anda juga harus bernama `404.html`.

3. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
4. Di daftar Bucket, pilih nama bucket yang ingin Anda gunakan untuk meng-host situs web statis.
5. Aktifkan hosting situs web statis untuk bucket Anda, lalu masukkan nama persis dokumen indeks Anda (misalnya, `404.html`). Untuk informasi selengkapnya, lihat Mengaktifkan hosting situs web dan Mengonfigurasi dokumen kesalahan khusus.

   Setelah mengaktifkan hosting situs web statis, lanjutkan ke langkah 6.

6. Untuk mengunggah dokumen kesalahan ke bucket Anda, lakukan salah satu hal berikut ini:

   - Seret dan jatuhkan file dokumen kesalahan ke dalam daftar bucket konsol.
   - Pilih Unggah, dan ikuti petunjuk untuk memilih dan mengunggah file indeks.

   Untuk step-by-step instruksi, lihatMengunggah Objek.

Langkah 8: Edit pengaturan Blokir Akses Publik S3

Dalam contoh ini, Anda mengedit pengaturan akses publik untuk bucket domain (`example.com`) untuk memungkinkan akses publik.

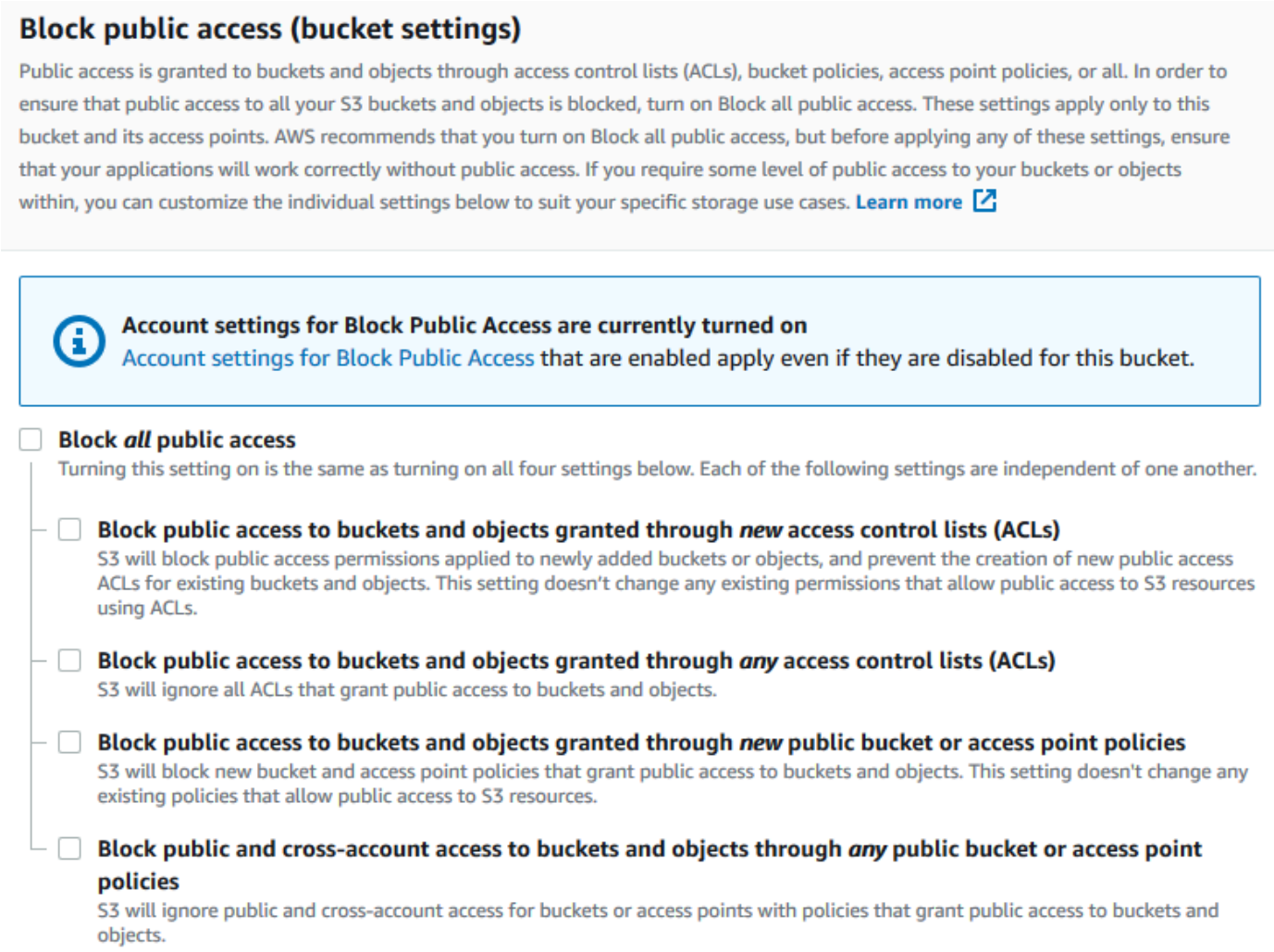
Secara default, Amazon S3 memblokir akses publik ke akun dan bucket Anda. Jika Anda ingin menggunakan bucket untuk menghosting situs web statis, Anda dapat menggunakan langkah-langkah ini untuk mengedit pengaturan blokir akses publik Anda.

> **Warning**
>
> Sebelum Anda menyelesaikan langkah ini, tinjau Melakukan blok akses publik ke penyimpanan Amazon S3 Anda untuk memastikan bahwa Anda telah memahami dan menerima risiko yang terkait dengan mengizinkan akses publik. Saat Anda mematikan pengaturan blokir akses publik untuk membuat bucket Anda menjadi publik, siapa pun di internet dapat mengakses bucket Anda. Kami sarankan agar Anda memblokir semua akses publik ke bucket Anda.

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pilih nama bucket yang telah Anda konfigurasikan sebagai situs web statis.
3. Pilih Izin.
4. Di bagian bawah Blokir akses publik (pengaturan bucket), pilih Edit.
5. Kosongkan Blokir semua akses publik, lalu pilih Simpan perubahan.

   > **Warning**
   >
   > Sebelum Anda menyelesaikan langkah ini, tinjau Melakukan blok akses publik ke penyimpanan Amazon S3 Anda untuk memastikan bahwa Anda telah memahami dan menerima risiko yang terkait dengan mengizinkan akses publik. Saat Anda mematikan

pengaturan blokir akses publik untuk membuat bucket Anda menjadi publik, siapa pun di internet dapat mengakses bucket Anda. Kami sarankan agar Anda memblokir semua akses publik ke bucket Anda.



Amazon S3 menonaktifkan pengaturan Blokir Akses Publik untuk bucket Anda. Untuk membuat situs web publik statis, Anda mungkin harus mengedit pengaturan Blokir Akses Publik untuk akun Anda sebelum menambahkan kebijakan bucket. Jika pengaturan akun untuk Blokir Akses Publik saat ini diaktifkan, Anda akan melihat catatan di Blokir akses publik (pengaturan bucket).

Langkah 9: Melampirkan kebijakan bucket

Dalam contoh ini, Anda melampirkan kebijakan bucket ke bucket domain (`example.com`) untuk memungkinkan akses baca publik. Anda mengganti *`Nama-Bucket`* dalam kebijakan bucket contoh dengan nama bucket domain Anda, misalnya `example.com`.

Setelah Anda mengedit pengaturan Blokir Akses Publik S3, Anda dapat menambahkan kebijakan bucket untuk memberikan akses baca publik ke bucket Anda. Saat Anda memberikan akses baca publik, siapa pun di internet dapat mengakses bucket Anda.

> **Important**
>
> Kebijakan berikut ini hanya merupakan contoh, dan memungkinkan akses penuh ke konten bucket Anda. Sebelum melanjutkan langkah ini, tinjau Bagaimana saya dapat mengamankan file dalam bucket Amazon S3 saya? untuk memastikan bahwa Anda telah memahami praktik terbaik untuk mengamankan file dalam bucket S3, dan risiko yang terlibat dalam pemberian akses publik.

1. Di bagian bawah Bucket, pilih nama bucket Anda.
2. Pilih Izin.
3. Di Bawah Kebijakan bucket, pilih Edit.
4. Untuk memberikan akses baca bagi publik untuk situs web Anda, salin kebijakan kelompok berikut, dan tempelkan di Editor kebijakan bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::Bucket-Name/*"
            ]
        }
    ]
```

```
}
```

5. Perbarui `Resource` dengan nama bucket Anda.

   Dalam contoh kebijakan bucket sebelumnya, *`Nama-Bucket`* adalah placeholder untuk nama bucket tersebut. Untuk menggunakan kebijakan bucket ini dengan bucket Anda sendiri, Anda harus memperbarui nama ini agar sesuai dengan nama bucket Anda.

6. Pilih Simpan perubahan.

   Pesan akan muncul, yang menunjukkan bahwa kebijakan bucket telah berhasil ditambahkan.

   Jika Anda melihat kesalahan yang mengatakan `Policy has invalid resource`, konfirmasikan bahwa nama bucket dalam kebijakan bucket tersebut sesuai dengan nama bucket Anda. Untuk informasi tentang menambahkan kebijakan bucket, lihat Bagaimana cara menambahkan kebijakan S3 bucket?

   Jika Anda mendapatkan pesan kesalahan dan tidak dapat menyimpan kebijakan bucket, periksa pengaturan akun dan bucket Blokir Akses Publik untuk mengonfirmasi bahwa Anda mengizinkan akses publik ke bucket.

Pada langkah berikutnya, Anda dapat mengetahui titik akhir situs web dan menguji titik akhir domain Anda.

Langkah 10: Menguji titik akhir domain Anda

Setelah Anda mengonfigurasikan bucket domain untuk menghosting situs web publik, Anda dapat menguji titik akhir. Untuk informasi selengkapnya, lihat Titik akhir situs web. Anda hanya dapat menguji titik akhir untuk bucket domain Anda, karena bucket subdomain Anda disiapkan untuk pengalihan situs web dan bukan hosting situs web statis.

> **Note**
>
> Amazon S3 tidak mendukung akses HTTPS ke situs web. Jika Anda ingin menggunakan HTTPS, Anda dapat menggunakan Amazon CloudFront untuk melayani situs web statis yang dihosting di Amazon S3.
> Untuk informasi selengkapnya, lihat Bagaimana CloudFront cara menggunakan situs web statis yang dihosting di Amazon S3? dan Memerlukan HTTPS untuk komunikasi antara pemirsa dan CloudFront.

1. Di bagian bawah Bucket, pilih nama bucket Anda.
2. Pilih Properti.
3. Di bagian bawah halaman, di bawah Hosting situs web statis, pilih Titik akhir situs web bucket.

   Dokumen indeks Anda terbuka di jendela browser terpisah.

Pada langkah berikutnya, Anda menggunakan Amazon Route 53 untuk memungkinkan pelanggan menggunakan kedua URL kustom Anda, agar pelanggan dapat menavigasi ke situs Anda.

Langkah 11: Menambahkan catatan alias untuk domain dan subdomain Anda

Pada langkah ini, Anda membuat catatan alias yang Anda tambahkan ke zona yang di-hosting untuk peta domain `example.com` dan `www.example.com` Anda. Alih-alih menggunakan alamat IP, catatan alias menggunakan titik akhir situs Amazon S3. Amazon Route 53 mempertahankan pemetaan antara catatan alias dan alamat IP tempat bucket Amazon S3 berada. Anda membuat dua rekaman alias, satu untuk domain root Anda, dan satu untuk subdomain Anda.

Menambahkan catatan alias untuk domain root dan subdomain Anda

Untuk menambahkan rekaman alias untuk domain akar Anda (**`example.com`**)

1. Buka konsol Route 53 di https://console.aws.amazon.com/route53/.

   > **Note**
   >
   > Jika Anda belum menggunakan Route 53, lihat Langkah 1: Mendaftarkan domain di Panduan Pengembang Amazon Route 53. Setelah menyelesaikan pengaturan Anda, Anda dapat melanjutkan petunjuknya.

2. Pilih Zona yang di-hosting.
3. Dalam daftar zona yang di-hosting, pilih nama zona yang di-hosting yang sesuai dengan nama domain Anda.
4. Pilih Buat catatan.
5. Pilih Beralih ke wizard.

> **Note**
>
> Jika Anda ingin menggunakan pembuatan cepat untuk membuat catatan alias, lihat Mengonfigurasi Route 53 untuk merutekan lalu lintas ke Bucket S3.

6. Pilih Perutean sederhana, dan pilih Selanjutnya.
7. Pilih Tentukan catatan sederhana.
8. Di Nama catatan, terima nilai defaultnya, yang merupakan nama zona yang di-hosting dan domain Anda.
9. Di Nilai/Rutekan lalu lintas ke, pilih Alias ke titik akhir situs web S3.
10. Pilih Wilayah.
11. Pilih bucket S3.

    Nama bucket harus sesuai dengan nama yang muncul di kotak Nama. Di daftar Pilih bucket S3, nama bucket muncul dengan titik akhir situs web Amazon S3 untuk Wilayah tempat bucket dibuat, misalnya, `s3-website-us-west-1.amazonaws.com (example.com)`.

    Pilih bucket S3 mencantumkan bucket, jika:

    - Anda telah mengonfigurasi bucket sebagai situs web statis.
    - Nama bucket sama dengan nama arsip yang Anda buat.
    - Arus Akun AWS menciptakan ember.

    Jika bucket Anda tidak muncul di daftar Pilih bucket S3, masukkan titik akhir situs web Amazon S3 untuk Wilayah tempat bucket dibuat, misalnya, **`s3-website-us-west-2.amazonaws.com`**. Untuk daftar lengkap titik akhir situs web Amazon S3, lihat Titik akhir Situs Web Amazon S3. Untuk informasi selengkapnya tentang target alias, lihat Nilai/rutekan lalu lintas ke di Panduan Pengembang Amazon Route 53.
12. Dalam jenis Rekam, pilih A - Rute lalu lintas ke alamat IPv4 dan beberapa AWS sumber daya.
13. Untuk Evaluasi kondisi target, pilih Tidak.
14. Pilih Tentukan catatan sederhana.

Untuk menambahkan rekaman alias untuk subdomain Anda (**`www.example.com`**)

1. Di bawah Konfigurasi catatan, pilih Tentukan catatan sederhana.

2. Di Nama catatan untuk subdomain Anda, ketik `www`.
3. Di Nilai/Rutekan lalu lintas ke, pilih Alias ke titik akhir situs web S3.
4. Pilih Wilayah.
5. Pilih bucket S3, misalnya, `s3-website-us-west-2.amazonaws.com` (`www.example.com`).

   Jika bucket Anda tidak muncul di daftar Pilih bucket S3, masukkan titik akhir situs web Amazon S3 untuk Wilayah tempat bucket dibuat, misalnya, **`s3-website-us-west-2.amazonaws.com`**. Untuk daftar lengkap titik akhir situs web Amazon S3, lihat Titik akhir Situs Web Amazon S3. Untuk informasi selengkapnya tentang target alias, lihat Nilai/rutekan lalu lintas ke di Panduan Pengembang Amazon Route 53.
6. Dalam jenis Rekam, pilih A - Rute lalu lintas ke alamat IPv4 dan beberapa AWS sumber daya.
7. Untuk Evaluasi kondisi target, pilih Tidak.
8. Pilih Tentukan catatan sederhana.
9. Pada halaman Konfigurasi catatan, pilih Buat catatan.

> **Note**
>
> Perubahan umumnya menyebar ke semua server nama Route 53 dalam waktu 60 detik. Setelah penyebarannya selesai, Anda dapat mengirimkan lalu lintas ke bucket Amazon S3 Anda dengan menggunakan nama rekaman alias yang Anda buat dalam prosedur ini.

Tambahkan rekaman alias untuk domain root dan subdomain Anda (konsol Route 53 yang lama)

Untuk menambahkan rekaman alias untuk domain root Anda (**`example.com`**)

Konsol Route 53 telah dirancang ulang. Di konsol Route 53, Anda dapat menggunakan konsol yang lama untuk sementara waktu. Jika Anda memilih untuk bekerja dengan konsol Route 53 yang lama, gunakan prosedur di bawah ini.

1. Buka konsol Route 53 di https://console.aws.amazon.com/route53/.

> **Note**
>
> Jika Anda belum menggunakan Route 53, lihat Langkah 1: Mendaftarkan domain di Panduan Pengembang Amazon Route 53. Setelah menyelesaikan pengaturan Anda, Anda dapat melanjutkan petunjuknya.

2. Pilih Zona yang Di-Hosting.
3. Dalam daftar zona yang di-hosting, pilih nama zona yang di-hosting yang sesuai dengan nama domain Anda.
4. Pilih Buat Set Catatan.
5. Tentukan nilai-nilai berikut ini:

   Nama

   Terima nilai default, yang merupakan nama zona yang di-hosting dan domain Anda.

   Untuk domain root, Anda tidak perlu memasukkan informasi tambahan apa pun di bidang Nama.

   Jenis

   Pilih A–Alamat IPv4.

   Alias

   Pilih Ya.

   Target Alias

   Di bagian Titik akhir situs web S3 dari daftar tersebut, pilih nama bucket Anda.

   Nama bucket harus sesuai dengan nama yang muncul di kotak Nama. Dalam daftar Target Alias, nama bucket diikuti oleh titik akhir situs web Amazon S3 untuk Wilayah tempat bucket dibuat, misalnya `example.com (s3-website-us-west-2.amazonaws.com)`. Target Alias membuat daftar bucket jika:

   - Anda telah mengonfigurasi bucket sebagai situs web statis.
   - Nama bucket sama dengan nama arsip yang Anda buat.
   - Arus Akun AWS menciptakan ember.

Jika bucket Anda tidak muncul di dalam daftar Target Alias, masukkan titik akhir situs web Amazon S3 untuk Wilayah tempat bucket dibuat, misalnya, `s3-website-us-west-2`. Untuk daftar lengkap titik akhir situs web Amazon S3, lihat Titik akhir Situs Web Amazon S3. Untuk informasi selengkapnya tentang target alias, lihat Nilai/rutekan lalu lintas ke di Panduan Pengembang Amazon Route 53.

Kebijakan Perutean

Terima nilai default Sederhana.

Mengevaluasi Kondisi Target

Terima nilai default Tidak.

6. Pilih Buat.

Untuk menambahkan rekaman alias untuk subdomain Anda (**`www.example.com`**)

1. Di zona yang di-hosting untuk domain root Anda (`example.com`), pilih Buat Set Catatan.
2. Tentukan nilai-nilai berikut ini:

   Nama

   Untuk subdomain, masukkan `www` di dalam kotak.

   Jenis

   Pilih A–Alamat IPv4.

   Alias

   Pilih Ya.

   Target Alias

   Di Titik akhir situs web S3 bagian dari daftar, pilih nama bucket yang sama dengan yang muncul di Nama bidang—misalnya, `www.example.com (s3-website-us-west-2.amazonaws.com)`.

   Kebijakan Perutean

   Terima nilai default Sederhana.

Mengevaluasi Kondisi Target

Terima nilai default Tidak.

3. Pilih Buat.

> **Note**
>
> Perubahan umumnya menyebar ke semua server nama Route 53 dalam waktu 60 detik. Setelah penyebarannya selesai, Anda dapat mengirimkan lalu lintas ke bucket Amazon S3 Anda dengan menggunakan nama rekaman alias yang Anda buat dalam prosedur ini.

Langkah 12: Menguji situs webnya

Verifikasi bahwa situs web dan pengalihannya bekerja dengan benar. Di browser, masukkan URL Anda. Dalam contoh ini, Anda dapat mencoba URL berikut ini:

- Domain (`http://example.com`)–Menampilkan dokumen indeks di bucket `example.com`.
- Subdomain (`http://www.example.com`)–Mengalihkan permintaan Anda ke `http://example.com`. Anda melihat dokumen indeks di bucket `example.com`.

Jika situs web Anda atau tautan pengalihan tidak berfungsi, Anda dapat mencoba hal berikut ini:

- Bersihkan cache–Menghapus cache browser web Anda.
- Periksa server nama–Jika halaman web dan tautan pengalihan tidak berfungsi setelah Anda mengosongkan cache, Anda dapat membandingkan server nama untuk domain Anda dan server nama untuk zona yang di-hosting. Jika server nama tidak cocok, Anda mungkin perlu memperbarui server nama domain Anda agar cocok dengan yang terdaftar di bawah zona yang di-hosting. Untuk informasi lebih lanjut, lihat Menambahkan atau mengubah server nama dan menyimpan glue record untuk domain.

Setelah Anda berhasil menguji domain root dan subdomain Anda, Anda dapat mengatur CloudFront distribusi Amazon untuk meningkatkan kinerja situs web Anda dan menyediakan log yang dapat Anda gunakan untuk meninjau lalu lintas situs web. Untuk informasi selengkapnya, lihat Mempercepat situs web Anda dengan Amazon CloudFront.

Mempercepat situs web Anda dengan Amazon CloudFront

Anda dapat menggunakan Amazon CloudFront untuk meningkatkan kinerja situs web Amazon S3 Anda. CloudFront membuat file situs web Anda (seperti HTML, gambar, dan video) tersedia dari pusat data di seluruh dunia (dikenal sebagai lokasi tepi). Ketika pengunjung meminta file dari situs web Anda, CloudFront secara otomatis mengalihkan permintaan ke salinan file di lokasi tepi terdekat. Ini menghasilkan waktu unduh yang lebih cepat daripada jika pengunjung meminta konten dari pusat data yang terletak lebih jauh.

CloudFront menyimpan konten di lokasi tepi untuk jangka waktu yang Anda tentukan. Jika pengunjung meminta konten yang telah di-cache lebih lama dari tanggal kedaluwarsa, CloudFront periksa server asal untuk melihat apakah versi konten yang lebih baru tersedia. Jika versi yang lebih baru tersedia, CloudFront salin versi baru ke lokasi tepi. Perubahan yang Anda lakukan pada konten asli direplikasi ke lokasi edge pada saat pengunjung meminta konten tersebut.

Menggunakan CloudFront tanpa Route 53

Tutorial di halaman ini menggunakan Route 53 untuk menunjuk ke CloudFront distribusi Anda. Namun, jika Anda ingin menayangkan konten yang dihosting di bucket Amazon S3 menggunakan CloudFront tanpa menggunakan Route 53, lihat CloudFrontTutorial Amazon: Menyiapkan Distribusi Konten Dinamis untuk Amazon S3. Saat menayangkan konten yang dihosting di bucket Amazon S3 menggunakan CloudFront, Anda dapat menggunakan nama bucket apa pun, dan HTTP dan HTTPS didukung.

Mengotomatisasi pengaturan dengan template AWS CloudFormation

Untuk informasi selengkapnya tentang menggunakan AWS CloudFormation templat untuk mengonfigurasi situs web statis aman yang membuat CloudFront distribusi untuk melayani situs web Anda, lihat Memulai situs web statis aman di Panduan CloudFront Pengembang Amazon.

Topik

- Langkah 1: Buat CloudFront distribusi
- Langkah 2: Memperbarui kumpulan catatan untuk domain dan subdomain Anda
- (Opsional) Langkah 3: Memeriksa file log

Langkah 1: Buat CloudFront distribusi

Pertama, Anda membuat CloudFront distribusi. Ini membuat situs web Anda tersedia dari pusat data di seluruh dunia.

Untuk membuat distribusi dengan asal Amazon S3

1. Buka CloudFront konsol dihttps://console.aws.amazon.com/cloudfront/v4/home.
2. Pilih Buat Distribusi.
3. Pada halaman Buat Distribusi, di bagian Pengaturan Asal, untuk Nama Domain Asal, masukkan titik akhir situs web Amazon S3 untuk bucket Anda—misalnya, **`example.com.s3-website.us-west-1.amazonaws.com`**.

   CloudFront mengisi ID Asal untuk Anda.
4. Untuk Pengaturan Perilaku Cache Default, pertahankan nilai yang diatur ke default.

   Dengan pengaturan default untuk Kebijakan Protokol Penampil, Anda dapat menggunakan HTTPS untuk situs web statis Anda. Untuk informasi selengkapnya opsi konfigurasi ini, lihat Nilai yang Anda Tentukan Saat Membuat atau Memperbarui Distribusi Web di Panduan CloudFront Pengembang Amazon.
5. Untuk Pengaturan Distribusi, lakukan hal berikut ini:

   a. Biarkan Kelas Harga agar diatur ke Gunakan Semua Lokasi Edge (Performa Terbaik).

   b. Tentukan Nama Domain Alternatif (CNAME) ke domain root dan subdomain `www`. Dalam tutorial ini, berikut merupakan `example.com` dan `www.example.com`.

   > **Important**
   >
   > Sebelum Anda melakukan langkah ini, perhatikan persyaratan untuk menggunakan nama domain alternatif, terutama terkait kebutuhan akan sertifikat SSL/TLS yang valid.

   c. Untuk Sertifikat SSL, pilih Sertifikat SSL Khusus (example.com), dan pilih sertifikat kustom yang mencakup domain dan nama subdomain.

   Untuk informasi selengkapnya, lihat Sertifikat SSL di Panduan CloudFront Pengembang Amazon.

   d. Di Objek Root Default, masukkan nama dokumen indeks Anda, misalnya, `index.html`.

   Jika URL yang digunakan untuk mengakses distribusi tidak berisi nama file, CloudFront distribusi mengembalikan dokumen indeks. Objek Root Default harus sama persis dengan nama dokumen indeks untuk situs web statis Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi dokumen indeks.

e. Tetapkan Pencatatan untuk Nyala.

> **Important**
>
> Saat Anda membuat atau memperbarui distribusi dan mengaktifkan CloudFront pencatatan, CloudFront perbarui daftar kontrol akses bucket (ACL) untuk memberikan `FULL_CONTROL` izin `awslogsdelivery` akun untuk menulis log ke bucket Anda. Untuk informasi selengkapnya, lihat Izin yang diperlukan untuk mengonfigurasi pencatatan standar dan mengakses file log Anda di Panduan CloudFront Pengembang Amazon. Jika bucket yang menyimpan log menggunakan setelan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek S3 untuk menonaktifkan ACL, CloudFront tidak dapat menulis log ke bucket. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

f. Untuk Bucket untuk Log, pilih bucket log yang Anda buat.

   Untuk informasi lebih lanjut tentang konfigurasi bucket log, lihat (Opsional) Mencatat lalu lintas web.

g. Jika Anda ingin menyimpan log yang dihasilkan oleh lalu lintas ke CloudFront distribusi dalam folder, di Awalan Log, masukkan nama folder.

h. Simpan semua pengaturan lain pada nilai defaultnya.

6. Pilih Buat Distribusi.
7. Untuk melihat status distribusi, cari distribusi di konsol dan periksa Status kolom.

   Status dari `InProgress` menunjukkan bahwa distribusi belum diterapkan sepenuhnya.

   Setelah distribusi Anda diterapkan, Anda dapat mereferensikan konten Anda dengan nama CloudFront domain baru.
8. Catat nilai Nama Domain yang ditampilkan di CloudFront konsol, misalnya,`dj4p1rv6mvubz.cloudfront.net`.
9. Untuk memverifikasi bahwa CloudFront distribusi Anda berfungsi, masukkan nama domain distribusi di browser web.

   Jika situs web Anda terlihat, CloudFront distribusi berfungsi. Jika situs web Anda memiliki domain khusus yang terdaftar di Amazon Route 53, Anda akan memerlukan nama CloudFront domain untuk memperbarui catatan yang ditetapkan pada langkah berikutnya.

Langkah 2: Memperbarui kumpulan catatan untuk domain dan subdomain Anda

Sekarang setelah Anda berhasil membuat CloudFront distribusi, perbarui catatan alias di Route 53 untuk menunjuk ke CloudFront distribusi baru.

Untuk memperbarui catatan alias untuk menunjuk ke distribusi CloudFront

1. Buka konsol Route 53 di https://console.aws.amazon.com/route53/.
2. Pada navigasi di sebelah kiri, pilih Zona yang di-hosting.
3. Pada halaman Zona yang Di-hosting, pilih zona yang di-hosting yang Anda buat untuk subdomain Anda, misalnya, `www.example.com`.
4. Di bagian bawah Catatan, pilih catatan A yang Anda buat untuk subdomain Anda.
5. Di bagian bawah Detail catatan, pilih Edit catatan.
6. Di bawah Rute lalu lintas ke, pilih Alias untuk CloudFront didistribusikan.
7. Di bawah Pilih distribusi, pilih CloudFront distribusi.
8. Pilih Simpan.
9. Untuk mengarahkan catatan A untuk domain root ke CloudFront distribusi, ulangi prosedur ini untuk domain root, misalnya,`example.com`.

   Pembaruan ke set catatan berlaku dalam 2–48 jam.
10. Untuk melihat apakah catatan A yang baru telah diterapkan, di browser web, masukkan URL subdomain Anda, misalnya, `http://www.example.com`.

    Jika browser tidak lagi mengarahkan Anda ke domain root (misalnya, `http://example.com`), catatan A yang baru tersedia. Ketika catatan A baru telah diterapkan, lalu lintas yang diarahkan oleh catatan A baru ke CloudFront distribusi tidak dialihkan ke domain root. Setiap pengunjung yang mereferensikan situs dengan menggunakan `http://example.com` atau `http://www.example.com` diarahkan ke lokasi CloudFront tepi terdekat, di mana mereka mendapat manfaat dari waktu pengunduhan yang lebih cepat.

    > **Tip**
    >
    > Browser dapat menyimpan pengaturan pengalihan. Jika menurut Anda pengaturan catatan A yang baru seharusnya sudah diterapkan, tetapi browser Anda masih mengalihkan `http://www.example.com` ke `http://example.com`, coba bersihkan

riwayat dan cache browser Anda, tutup dan buka kembali aplikasi browser Anda, atau gunakan browser web yang berbeda.

(Opsional) Langkah 3: Memeriksa file log

Log akses memberi tahu Anda berapa banyak orang yang mengunjungi situs web. Cookie ini juga berisi data bisnis berharga yang dapat Anda analisis dengan layanan lain, seperti Amazon EMR.

CloudFront log disimpan di bucket dan folder yang Anda pilih saat Anda membuat CloudFront distribusi dan mengaktifkan logging. CloudFront menulis log ke bucket log Anda dalam waktu 24 jam sejak permintaan terkait dibuat.

Untuk melihat file log situs web Anda

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pilih nama bucket pencatatan log untuk situs web Anda.
3. Pilih folder CloudFront log.
4. Unduh `.gzip` file yang ditulis oleh CloudFront sebelum membukanya.

   Jika Anda membuat situs web hanya sebagai latihan pembelajaran, Anda dapat menghapus sumber daya yang Anda alokasikan, sehingga Anda tidak lagi dikenakan biaya. Untuk melakukannya, lihat Membersihkan sumber daya contoh Anda. Setelah Anda menghapus sumber daya AWS , situs web Anda tidak lagi tersedia.

Membersihkan sumber daya contoh Anda

Jika Anda membuat situs web statis hanya sebagai latihan pembelajaran, hapus sumber daya AWS yang Anda alokasikan, sehingga Anda tidak lagi dikenakan biaya. Setelah Anda menghapus sumber daya AWS , situs web Anda tidak lagi tersedia.

Tugas

- Langkah 1: Hapus CloudFront distribusi Amazon
- Langkah 2: Menghapus Route 53 zona yang di-hosting
- Langkah 3: Menonaktifkan pencatatan dan hapus bucket S3 Anda

Langkah 1: Hapus CloudFront distribusi Amazon

Sebelum Anda menghapus CloudFront distribusi Amazon, Anda harus menonaktifkannya. Distribusi yang dinonaktifkan tidak lagi berfungsi, dan tidak dikenakan biaya. Anda dapat mengaktifkan distribusi yang dinonaktifkan kapan saja. Setelah Anda menghapus distribusi yang dinonaktifkan, distribusi tersebut tidak lagi tersedia.

Untuk menonaktifkan dan menghapus CloudFront distribusi

1. Buka CloudFront konsol dihttps://console.aws.amazon.com/cloudfront/v4/home.
2. Pilih distribusi yang ingin Anda nonaktifkan, lalu pilih Nonaktifkan.
3. Saat diminta untuk mengonfirmasi, pilih Ya, Nonaktifkan.
4. Pilih distribusi yang dinonaktifkan, lalu pilih Hapus.
5. Saat diminta konfirmasi, pilih Ya, Hapus.

Langkah 2: Menghapus Route 53 zona yang di-hosting

Sebelum menghapus zona yang di-hosting, Anda harus menghapus set rekaman yang Anda buat. Anda tidak perlu menghapus catatan NS dan SOA; catatan ini secara otomatis dihapus ketika Anda menghapus zona yang di-hosting.

Untuk menghapus set catatan

1. Buka konsol Route 53 di https://console.aws.amazon.com/route53/.
2. Di dalam daftar nama domain, pilih nama domain Anda, lalu pilih Masuk ke Set Catatan.
3. Dalam daftar kumpulan rekaman, pilih rekaman A yang Anda buat.

   Jenis setiap set catatan tercantum dalam Jenis kolom.
4. Pilih Hapus Set Catatan.
5. Ketika diminta untuk mengonfirmasi, pilih Konfirmasi.

Untuk menghapus zona Route 53 yang di-hosting

1. Lanjutkan dari prosedur sebelumnya, pilih Kembali ke Zona yang Di-hosting.
2. Pilih nama domain Anda, lalu pilih Hapus Zona yang Di-hosting.
3. Ketika diminta untuk mengonfirmasi, pilih Konfirmasi.

Langkah 3: Menonaktifkan pencatatan dan hapus bucket S3 Anda

Sebelum Anda menghapus bucket S3 Anda, pastikan pembuatan log dinonaktifkan untuk bucket. Jika tidak, AWS terus menulis log ke bucket Anda saat Anda menghapusnya.

Untuk menonaktifkan pembuatan log untuk bucket

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di bagian bawah Bucket, pilih nama bucket Anda, lalu pilih Properti.
3. Dari Properti, pilih Pencatatan.
4. Hapus kotak centang Diaktifkan.
5. Pilih Simpan.

Sekarang, Anda dapat menghapus bucket Anda. Untuk informasi selengkapnya, lihat Menghapus bucket.

Membuat, mengonfigurasi, dan bekerja dengan bucket Amazon S3

Untuk menyimpan data Anda di Amazon S3, Anda bekerja dengan sumber daya yang dikenal sebagai bucket dan objek. Bucket adalah kontainer untuk objek. Objek adalah file data dan metadata apa pun yang mendeskripsikan file tersebut.

Untuk menyimpan objek di Amazon S3, Anda membuat bucket dan kemudian mengunggah objek tersebut ke dalam bucket. Bila objek ada di dalam bucket, Anda bisa membuka, mengunduh, dan memindahkannya. Saat Anda tidak lagi memerlukan objek atau bucket, Anda dapat membersihkan sumber daya Anda.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

> **Note**
>
> Dengan Amazon S3, Anda hanya membayar sesuai penggunaan Anda. Untuk informasi selengkapnya tentang fitur dan harga Amazon S3, lihat Amazon S3. Jika Anda adalah pelanggan baru Amazon S3, Anda dapat memulai Amazon S3 secara gratis. Untuk informasi selengkapnya, lihat AWS Tingkat Gratis.

Topik dalam bagian ini memberikan gambaran umum tentang bekerja dengan bucket di Amazon S3. Termasuk informasi tentang penamaan, pembuatan, akses, dan penghapusan bucket. Untuk informasi selengkapnya tentang melihat atau mencantumkan objek dalam bucket, lihat Mengatur, membuat daftar, dan bekerja dengan objek Anda.

Topik

- Gambaran umum bucket
- Peraturan penamaan bucket
- Mengakses dan mendaftarkan bucket Amazon S3

- Membuat bucket
- Melihat properti untuk bucket S3
- Mengosongkan bucket
- Menghapus bucket
- Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3
- Bekerja dengan Mountpoint untuk Amazon S3
- Mengonfigurasi transfer file yang cepat dan aman menggunakan Amazon S3 Transfer Acceleration
- Menggunakan bucket Pembayaran Pemohon untuk transfer dan penggunaan penyimpanan
- Pembatasan dan batasan bucket

Gambaran umum bucket

Untuk mengunggah data Anda (foto, video, dokumen, dll.) ke Amazon S3, Anda harus terlebih dahulu membuat bucket S3 di salah satu Wilayah AWS.

Bucket adalah kontainer untuk objek yang disimpan dalam Amazon S3. Anda dapat menyimpan berapa pun jumlah objek dalam bucket dan dapat memiliki hingga 100 bucket di akun Anda. Untuk meminta peningkatan, kunjungi Konsol Kuota Layanan.

Setiap objek dimuat dalam bucket. Misalnya, jika objek bernama `photos/puppy.jpg` disimpan dalam bucket `DOC-EXAMPLE-BUCKET` di Wilayah AS Barat (Oregon), maka objek tersebut dapat dialamatkan dengan menggunakan URL `https://DOC-EXAMPLE-BUCKET.s3.us-west-2.amazonaws.com/photos/puppy.jpg`. Untuk informasi lebih lanjut, lihat Mengakses Bucket.

Dalam hal implementasi, bucket dan objek adalah AWS sumber daya, dan Amazon S3 menyediakan API bagi Anda untuk mengelolanya. Misalnya, Anda dapat membuat bucket serta mengunggah objek menggunakan API Amazon S3. Anda juga dapat menggunakan konsol Amazon S3 untuk melakukan operasi ini. Konsol menggunakan API Amazon S3 untuk mengirim permintaan ke Amazon S3.

Bagian ini menjelaskan cara untuk bekerja dengan bucket. Untuk informasi tentang bekerja dengan objek, lihat Gambaran umum objek Amazon S3.

Amazon S3 mendukung bucket global, yang berarti bahwa setiap nama bucket harus unik di semua bagian Akun AWS dalam partisi Wilayah AWS . Partisi adalah pembuatan grup Wilayah. Saat ini, AWS memiliki tiga partisi: `aws` (Wilayah Standar), `aws-cn` (Wilayah Tiongkok), dan `aws-us-gov` (AWS GovCloud (US)).

Setelah bucket dibuat, nama bucket tersebut tidak dapat digunakan oleh yang lain Akun AWS di partisi yang sama sampai bucket dihapus. Anda tidak boleh bergantung pada konvensi penamaan bucket kustom untuk tujuan ketersediaan, atau verifikasi keamanan. Untuk panduan penamaan bucket, lihat Peraturan penamaan bucket.

Amazon S3 membuat bucket di Wilayah yang Anda tentukan. Untuk mengurangi latensi, meminimalkan biaya, atau memenuhi persyaratan peraturan, pilih Wilayah AWS yang secara geografis dekat dengan Anda. Misalnya, jika Anda tinggal di Eropa, Anda mungkin akan mendapatkan keuntungan jika membuat bucket di Wilayah Eropa (Irlandia) atau Eropa (Frankfurt). Untuk daftar semua Wilayah dan titik akhir Amazon S3, lihat Wilayah dan Titik Akhir dalam AWS Referensi Umum.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

> **Note**
>
> Objek yang termasuk dalam bucket yang Anda buat di spesifik Wilayah AWS tidak pernah meninggalkan Wilayah itu, kecuali jika Anda secara eksplisit mentransfernya ke Wilayah lain. Misalnya, objek yang disimpan di Wilayah Eropa (Irlandia) tidak pernah keluar dari Wilayah tersebut.

Topik

- Tentang izin
- Mengelola akses publik ke bucket
- Opsi konfigurasi bucket

Tentang izin

Anda dapat menggunakan Pengguna root akun AWS kredensyal Anda untuk membuat bucket dan melakukan operasi Amazon S3 lainnya. Namun, kami menyarankan agar Anda tidak menggunakan kredensyal pengguna root Anda Akun AWS untuk membuat permintaan, seperti membuat bucket.

Sebagai gantinya, buat pengguna AWS Identity and Access Management (IAM), dan berikan akses penuh kepada pengguna tersebut (pengguna secara default tidak memiliki izin).

Pengguna ini disebut sebagai administrator. Anda dapat menggunakan kredensil pengguna administrator, bukan kredenal pengguna root akun Anda, untuk berinteraksi AWS dan melakukan tugas, seperti membuat bucket, membuat pengguna, dan memberi mereka izin.

Untuk informasi lebih lanjut, lihat kredensial Pengguna root akun AWS dan kredensial pengguna IAM dalam AWS Referensi Umum dan Praktik terbaik keamanan di IAM dalam Panduan Pengguna IAM.

Akun AWS Yang menciptakan sumber daya memiliki sumber daya itu. Misalnya, jika Anda membuat pengguna IAM di dalam Akun AWS dan memberikan izin kepada pengguna untuk membuat bucket, pengguna dapat membuat bucket. Tetapi pengguna tidak memiliki ember; Akun AWS yang dimiliki pengguna memiliki ember. Pengguna memerlukan izin tambahan dari pemilik sumber daya untuk melakukan operasi bucket lainnya. Untuk informasi selengkapnya tentang mengelola izin untuk sumber daya Amazon S3, lihat Identity and Access Management untuk Amazon S3.

Mengelola akses publik ke bucket

Akses publik diberikan kepada bucket dan objek melalui daftar kontrol akses (ACL), kebijakan bucket, atau keduanya. Untuk membantu Anda mengelola akses publik ke sumber daya Amazon S3, Amazon S3 menyediakan pengaturan untuk memblokir akses publik. Pengaturan Blokir Akses Publik Amazon S3 dapat menggantikan ACL dan kebijakan bucket, sehingga Anda dapat memberlakukan batas yang sama pada akses publik ke sumber daya ini. Anda dapat menerapkan pengaturan Blokir Akses Publik ke bucket individu, atau semua bucket dalam akun Anda.

Untuk memastikan bahwa semua bucket dan objek Amazon S3 Anda memblokir akses publiknya, keempat pengaturan untuk Blokir Akses Publik diaktifkan secara default saat Anda membuat bucket baru. Kami merekomendasikan agar Anda mengaktifkan keempat pengaturan untuk Blokir Akses Publik untuk akun Anda juga. Pengaturan ini memblokir semua akses publik untuk semua bucket saat ini, dan yang akan datang.

Sebelum menerapkan pengaturan ini, verifikasi bahwa aplikasi Anda akan bekerja dengan baik tanpa akses publik. Jika Anda memerlukan beberapa tingkat akses publik ke bucket atau objek Anda—misalnya, untuk menjadi host situs web statis seperti yang dijelaskan di Hosting situs web statis menggunakan Amazon S3—Anda dapat menyesuaikan pengaturan individual agar sesuai dengan kasus penggunaan penyimpanan Anda. Untuk informasi selengkapnya, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

Namun, kami menyarankan agar Anda membiarkan Blokir Akses Publik diaktifkan. Jika Anda ingin mengaktifkan keempat pengaturan Blokir Akses Publik dan meng-host situs web statis, Anda dapat menggunakan Amazon CloudFront Origin Access Control (OAC). Amazon CloudFront menyediakan kemampuan yang diperlukan untuk menyiapkan situs web statis yang aman. Situs web statis Amazon S3 hanya mendukung titik akhir HTTP. Amazon CloudFront menggunakan penyimpanan Amazon S3 yang tahan lama sambil menyediakan header keamanan tambahan, seperti HTTPS. HTTPS menambahkan keamanan dengan mengenkripsi permintaan HTTP normal, dan melindungi dari serangan siber umum.

Untuk informasi selengkapnya, lihat Memulai situs web statis aman di Panduan CloudFront Pengembang Amazon.

> **Note**
>
> Jika Anda melihat `Error` saat Anda mencantumkan bucket dan pengaturan akses publiknya, Anda mungkin tidak memiliki izin yang diperlukan. Pastikan bahwa Anda menambahkan izin berikut ke kebijakan pengguna atau peran Anda:
>
> ```
> s3:GetAccountPublicAccessBlock
> s3:GetBucketPublicAccessBlock
> s3:GetBucketPolicyStatus
> s3:GetBucketLocation
> s3:GetBucketAcl
> s3:ListAccessPoints
> s3:ListAllMyBuckets
> ```
>
> Dalam beberapa kasus yang jarang terjadi, permintaan juga dapat gagal karena Wilayah AWS gangguan.

Opsi konfigurasi bucket

Amazon S3 mendukung berbagai opsi bagi Anda untuk mengonfigurasi bucket Anda. Misalnya, Anda dapat mengonfigurasi bucket untuk hosting situs web, menambahkan konfigurasi untuk mengelola siklus hidup objek di bucket, dan mengonfigurasi bucket untuk mencatat semua akses ke bucket. Amazon S3 mendukung sub-sumber daya untuk menyimpan dan mengelola informasi konfigurasi bucket. Anda dapat menggunakan API Amazon S3 untuk membuat dan mengelola sub-sumber daya ini. Namun, Anda juga dapat menggunakan konsol atau AWS SDK.

> **Note**
>
> Terdapat konfigurasi tingkat objek juga. Misalnya, Anda dapat mengonfigurasi izin tingkat objek dengan mengonfigurasi daftar kontrol akses (ACL) kustom untuk objek tersebut.

Ini disebut sebagai sub-sumber daya karena ada dalam konteks bucket atau objek tertentu. Tabel berikut ini mencantumkan sub-sumber daya yang memungkinkan Anda mengelola konfigurasi kustom bucket.

Sub-sumber daya	Deskripsi
cors (berbagi sumber daya lintas asal)	Anda dapat mengonfigurasi bucket Anda untuk mengizinkan permintaan lintas asal. Untuk informasi selengkapnya, lihat Berbagi sumber daya lintas asal (CORS).
pemberitahuan peristiwa	Anda dapat mengaktifkan bucket Anda untuk mengirimkan pemberitahuan peristiwa bucket tertentu. Untuk informasi selengkapnya, lihat Notifikasi Peristiwa Amazon S3.
siklus hidup	Anda dapat menentukan aturan siklus aktif objek dalam bucket Anda, yang memiliki siklus aktif yang ditetapkan dengan baik. Misalnya, Anda dapat menetapkan aturan untuk mengarsipkan objek satu tahun setelah pembuatan , atau menghapus objek 10 tahun setelah pembuatan. Untuk informasi selengkapnya, lihat Mengelola siklus hidup penyimpanan Anda.
lokasi	Saat membuat bucket, Anda menentukan Wilayah AWS tempat Amazon S3 yang Anda inginkan untuk membuat bucket. Amazon S3 menyimpan informasi ini di sub-sumber daya lokasi dan menyediakan API bagi Anda untuk mengambil informasi ini.
pencatatan log	Pencatatan log memungkinkan Anda untuk melacak permintaan akses ke bucket Anda. Setiap catatan log akses memberikan perincian tentang permintaan akses tunggal, seperti pemohon, nama bucket, waktu permintaa

Sub-sumber daya	Deskripsi
	n, tindakan permintaan, status respons, dan kode kesalahan, jika ada. Informasi log akses dapat berguna dalam audit keamanan dan akses. Ini juga dapat membantu Anda untuk mempelajari basis pelanggan Anda, serta memahami tagihan Amazon S3. Untuk informasi selengkapnya, lihat Pencatatan permintaan dengan pencatatan akses server.
penguncian objek	Untuk menggunakan Kunci Objek S3, Anda harus mengaktifkannya untuk bucket. Anda juga dapat secara opsional mengonfigurasi mode dan periode retensi default yang berlaku untuk objek baru yang ditempatkan dalam bucket. Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek S3.
kebijakan dan ACL (daftar kontrol akses)	Semua sumber daya Anda (seperti bucket dan objek) bersifat pribadi secara default. Amazon S3 mendukung opsi kebijakan bucket dan daftar kontrol akses (ACL) bagi Anda untuk memberikan dan mengelola izin tingkat bucket. Amazon S3 menyimpan informasi izin di sub-sumber daya kebijakan dan acl. Untuk informasi selengkapnya, lihat Identity and Access Management untuk Amazon S3.
replikasi	Replikasi adalah penyalinan objek secara otomatis dan asinkron di seluruh bucket di Wilayah AWS yang berbeda atau sama. Untuk informasi selengkap nya, lihat Mereplikasi ikhtisar objek.
requestPayment	Secara default, Akun AWS yang membuat bucket (pemilik bucket) membayar unduhan dari bucket. Dengan menggunakan sub-sumber daya ini, pemilik bucket dapat menetapkan bahwa orang yang meminta unduhan akan dikenakan biaya untuk pengunduhan. Amazon S3 menyediakan API bagi Anda untuk dapat mengelola sub-sumber daya ini. Untuk informasi selengkapnya, lihat Menggunakan bucket Pembayaran Pemohon untuk transfer dan penggunaan penyimpanan.

Sub-sumber daya	Deskripsi
penandaan	Anda dapat menambahkan tag alokasi biaya ke bucket Anda untuk mengkategorikan dan melacak biaya Anda. AWS Amazon S3 menyediakan sub-sumber daya pemberian tag untuk menyimpan dan mengelola tag di bucket. Menggunakan tag yang Anda terapkan ke bucket, AWS buat laporan alokasi biaya dengan penggunaan dan biaya yang dikumpulkan oleh tag Anda. Untuk informasi selengkapnya, lihat Pelaporan penagihan dan penggunaan untuk Amazon S3.
akselerasi transfer	Akselerasi Transfer memungkinkan transfer file yang cepat, mudah, dan aman dalam jarak jauh antara klien Anda dan bucket S3. Transfer Acceleration memanfaatkan lokasi tepi Amazon yang didistribusikan secara global CloudFront. Untuk informasi selengkapnya, lihat Mengonfigurasi transfer file yang cepat dan aman menggunakan Amazon S3 Transfer Acceleration.
Penentuan Versi	Penentuan Versi membantu Anda memulihkan penghapusan dan penimpaan secara tidak sengaja. Kami merekomendasikan Penentuan Versi sebagai praktik terbaik untuk memulihkan objek agar tidak dihapus atau ditimpa secara tidak sengaja. Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.
situs web	Anda dapat mengonfigurasi bucket untuk menghosting situs web statis. Amazon S3 menyimpan konfigurasi ini dengan membuat sub-sumber daya situs web. Untuk informasi selengkapnya, lihat Hosting situs web statis menggunakan Amazon S3.

Peraturan penamaan bucket

Aturan berikut berlaku untuk penamaan bucket tujuan umum dan bucket direktori di Amazon S3:

Topik

- Aturan penamaan bucket tujuan umum
- Aturan penamaan bucket direktori

Aturan penamaan bucket tujuan umum

Aturan penamaan berikut berlaku untuk bucket tujuan umum.

- Panjang nama bucket harus antara 3 (menit) dan 63 (maks) karakter.
- Nama bucket hanya bisa terdiri dari huruf kecil, angka, titik (.), dan tanda hubung (-).
- Nama bucket harus diawali dan juga diakhiri dengan huruf atau, nomor.
- Nama bucket tidak boleh berisi dua periode yang berdekatan.
- Nama bucket tidak boleh diformat sebagai alamat IP (misalnya, 192.168.5.4).
- Nama bucket tidak boleh dimulai dengan prefiks `xn--`.
- Nama bucket tidak boleh dimulai dengan awalan `sthree-` atau awalan`sthree-configurator`.
- Nama bucket tidak boleh diakhiri dengan sufiks `-s3alias`. Sufiks ini dicadangkan untuk nama alias titik akses. Untuk informasi selengkapnya, lihat Menggunakan alias gaya bucket untuk titik akses bucket S3 Anda.
- Nama bucket tidak boleh diakhiri dengan sufiks `--ol-s3`. Sufiks ini dicadangkan untuk nama alias Titik Akses Lambda Objek. Untuk informasi selengkapnya, lihat Cara menggunakan alias gaya bucket untuk bucket S3 Anda Titik Akses Lambda Objek.
- Nama bucket harus unik Akun AWS di semua bagian Wilayah AWS dalam partisi. Partisi adalah pengelompokan Wilayah. AWS saat ini memiliki tiga partisi: `aws` (Wilayah Standar), `aws-cn` (Wilayah China), dan `aws-us-gov` (AWS GovCloud (US)).
- Nama bucket tidak dapat digunakan oleh orang lain Akun AWS di partisi yang sama sampai bucket dihapus.
- Bucket yang digunakan dengan Amazon S3 Transfer Acceleration tidak memiliki titik (.) pada namanya. Untuk informasi lebih lanjut tentang Transfer Acceleration, lihat Mengonfigurasi transfer file yang cepat dan aman menggunakan Amazon S3 Transfer Acceleration.

Untuk kompatibilitas terbaik, kami menyarankan agar Anda menghindari penggunaan titik (.) dalam nama bucket, kecuali untuk bucket yang digunakan hanya untuk menghosting situs web statis. Jika Anda menyertakan titik dalam nama bucket, Anda tidak dapat menggunakan virtual-host-style pengalamatan melalui HTTPS, kecuali Anda melakukan validasi sertifikat Anda sendiri. Hal ini dikarenakan sertifikat keamanan yang digunakan untuk hosting virtual bucket tidak berfungsi untuk bucket yang memiliki titik pada namanya.

Batasan ini tidak memengaruhi bucket yang digunakan untuk hosting situs web statis, karena hosting situs web statis hanya tersedia melalui HTTP. Untuk informasi lebih lanjut tentang virtual-host-style pengalamatan, lihatBucket dengan hosting virtual. Untuk informasi tentang hosting situs web, lihat Hosting situs web statis menggunakan Amazon S3.

> **Note**
>
> Sebelum 1 Maret 2018, bucket yang dibuat di Wilayah AS Timur (Virginia Utara) dapat memiliki nama hingga mencapai 255 karakter, dan menyertakan huruf besar dan garis bawah. Mulai 1 Maret 2018, bucket baru di AS Timur (Virginia Utara) harus mematuhi aturan yang sama yang berlaku di semua Wilayah lainnya.

Untuk informasi tentang nama kunci objek, lihat Membuat nama kunci objek.

Contoh nama bucket tujuan umum

Contoh nama bucket berikut ini bersifat valid, dan mengikuti panduan penamaan yang disarankan untuk bucket tujuan umum:

- `docexamplebucket1`
- `log-delivery-march-2020`
- `my-hosted-content`

Contoh nama bucket berikut ini bersifat valid, tetapi tidak disarankan untuk penggunaan selain hosting situs web statis:

- `docexamplewebsite.com`
- `www.docexamplewebsite.com`
- `my.example.s3.bucket`

Contoh nama bucket berikut bersifat tidak valid:

- `doc_example_bucket` (mengandung garis bawah)
- `DocExampleBucket` (mengandung huruf besar)
- `doc-example-bucket-` (berakhir dengan tanda hubung)

Aturan penamaan bucket direktori

Nama bucket direktori harus:

- Jadilah unik dalam Zona yang dipilih Wilayah AWS dan Availability Zone.
- Panjang nama harus antara 3 (menit) dan 63 (maks) karakter, termasuk akhiran.
- Hanya terdiri dari huruf kecil, angka, tanda hubung (-).
- Dimulai dan diakhiri dengan huruf atau angka.
- Harus menyertakan akhiran berikut:`--azid--x-s3`.

> **Note**
>
> Saat Anda membuat bucket direktori menggunakan konsol, akhiran secara otomatis ditambahkan ke nama dasar yang Anda berikan. Sufiks ini mencakup ID Zona Ketersediaan dari Zona Ketersediaan yang Anda pilih.
> Saat membuat bucket direktori menggunakan API, Anda harus memberikan akhiran lengkap, termasuk ID Availability Zone, dalam permintaan Anda. Untuk daftar ID Availability Zone, lihatZona Ketersediaan dan Wilayah S3 Express One Zone.

Mengakses dan mendaftarkan bucket Amazon S3

Untuk membuat daftar dan mengakses bucket Amazon S3 Anda, Anda dapat menggunakan berbagai macam alat. Tinjau alat-alat berikut ini untuk menentukan pendekatan mana yang sesuai dengan kasus penggunaan Anda:

- Konsol Amazon S3: Dengan konsol Amazon S3, Anda dapat dengan mudah mengakses bucket dan memodifikasi properti bucket. Anda juga dapat melakukan sebagian besar operasi bucket menggunakan UI konsol, tanpa harus menulis kode apa pun.

- AWS CLI: Jika Anda perlu mengakses beberapa bucket, Anda dapat menghemat waktu dengan menggunakan AWS Command Line Interface (AWS CLI) untuk mengotomatiskan tugas umum dan berulang. Kemampuan naskah dan pengulangan untuk tindakan umum sering menjadi pertimbangan seiring bertumbuhnya organisasi. Untuk informasi selengkapnya, lihat Mengembangkan dengan Amazon S3 menggunakan AWS CLI.
- API REST Amazon S3: Anda dapat menggunakan API REST Amazon S3 untuk menulis program Anda sendiri dan mengakses bucket secara terprogram. Amazon S3 mendukung arsitektur API di mana bucket dan objek Anda adalah sumber dayanya, masing-masing dengan URI sumber daya yang secara unik mengidentifikasi sumber daya. Untuk informasi selengkapnya, lihat Berkembang dengan Amazon S3 menggunakan API REST.

Bergantung pada kasus penggunaan bucket Amazon S3 Anda, ada berbagai metode yang disarankan untuk mengakses data yang mendasarinya di bucket Anda. Daftar berikut mencakup kasus penggunaan umum untuk mengakses data Anda.

- Situs web statis–Anda dapat menggunakan Amazon S3 untuk host situs web statis. Dalam kasus penggunaan ini, Anda dapat mengonfigurasi bucket S3 agar berfungsi seperti situs web. Untuk contoh yang memandu Anda melalui langkah-langkah menghosting situs web di Amazon S3, lihat Tutorial: Mengonfigurasi situs web statis untuk Amazon S3.

  Untuk meng-host situs web statis dengan pengaturan keamanan seperti Blokir Akses Publik diaktifkan, sebaiknya gunakan Amazon CloudFront dengan Origin Access Control (OAC) dan menerapkan header keamanan tambahan, seperti HTTPS. Untuk informasi selengkapnya, lihat Memulai dengan situs web statis yang aman.

  > **Note**
  >
  > Amazon S3 mendukung URL yang dihosting bergaya virtual dan bergaya jalur untuk akses situs web statis. Karena bucket dapat diakses menggunakan URL bergaya jalur dan yang dihosting virtual, kami menyarankan agar Anda membuat bucket dengan nama bucket yang sesuai dengan DNS. Untuk informasi selengkapnya, lihat Pembatasan dan batasan bucket.

- Set data bersama–Saat Anda menskalakan di Amazon S3, mengadopsi model multi-penyewa adalah hal yang umum, di mana Anda menetapkan pelanggan akhir atau unit bisnis yang berbeda ke prefiks unik dalam bucket bersama. Dengan menggunakan Titik akses Amazon S3, Anda dapat membagi satu kebijakan bucket besar menjadi kebijakan titik akses terpisah dan terpisah untuk

setiap aplikasi yang perlu mengakses set data bersama. Pendekatan ini membuatnya lebih mudah untuk fokus pada membangun kebijakan akses yang tepat untuk aplikasi tanpa mengganggu apa yang dilakukan aplikasi lain dalam set data bersama. Untuk informasi selengkapnya, lihat Mengelola akses data dengan titik akses Amazon S3.

- Beban kerja throughput tinggi–Mountpoint untuk Amazon S3 adalah klien file open source throughput tinggi untuk memasang bucket Amazon S3 sebagai sistem file lokal. Dengan Mountpoint, aplikasi Anda dapat mengakses objek yang disimpan di Amazon S3 melalui operasi sistem file, seperti buka dan baca. Mountpoint secara otomatis menerjemahkan operasi ini ke dalam panggilan API objek S3, memberikan aplikasi Anda akses ke penyimpanan elastis serta throughput Amazon S3 melalui antarmuka file. Untuk informasi selengkapnya, lihat Bekerja dengan Mountpoint untuk Amazon S3.
- Aplikasi Multi-Wilayah—Titik Akses Multi-Wilayah Amazon S3 menyediakan titik akhir global yang dapat digunakan aplikasi untuk memenuhi permintaan dari bucket S3 yang terletak di beberapa Wilayah AWS. Anda dapat menggunakan Titik Akses Multi-Wilayah untuk membangun aplikasi multi-Wilayah dengan arsitektur yang sama dengan yang digunakan di satu Wilayah, dan kemudian menjalankan aplikasi tersebut di mana saja di seluruh dunia. Alih-alih mengirim permintaan melalui internet publik, Titik Akses Multi-Wilayah menyediakan ketahanan jaringan bawaan dengan percepatan permintaan berbasis internet ke Amazon S3. Untuk informasi selengkapnya, lihat Titik Akses Multi-Wilayah di Amazon S3.
- Membangun aplikasi baru — Anda dapat menggunakan AWS SDK saat mengembangkan aplikasi dengan Amazon S3. AWS SDK menyederhanakan tugas pemrograman Anda dengan membungkus API Amazon S3 REST yang mendasarinya. Untuk membangun aplikasi seluler dan web yang terhubung, Anda dapat menggunakan SDK AWS Seluler dan AWS Amplify JavaScript perpustakaan. Untuk informasi selengkapnya, lihat Mengembangkan dengan Amazon S3 menggunakan SDK AWS.
- Secure Shell (SSH) File Transfer Protocol (SFTP) — Jika Anda mencoba mentransfer data sensitif melalui internet dengan aman, Anda dapat menggunakan server berkemampuan SFTP dengan bucket Amazon S3 Anda. AWS SFTP adalah protokol jaringan yang mendukung fungsionalitas keamanan dan otentikasi penuh SSH. Dengan protokol ini, Anda memiliki kontrol halus atas identitas pengguna, izin, dan kunci. Atau, Anda dapat menggunakan kebijakan IAM untuk mengelola akses. Untuk mengaitkan server berkemampuan SFTP dengan bucket Amazon S3 Anda, pastikan untuk membuat server berkemampuan SFTP terlebih dahulu. Kemudian, Anda mengatur akun pengguna, dan mengaitkan server dengan bucket Amazon S3. Untuk panduan proses ini, lihat AWS Transfer for SFTP - Layanan SFTP yang Dikelola Sepenuhnya untuk Amazon S3 di Blog.AWS

Mendaftarkan bucket

Untuk mendaftarkan semua bucket, Anda harus memiliki izin `s3:ListAllMyBuckets`. Untuk mengakses bucket, pastikan juga mendapatkan izin yang diperlukan AWS Identity and Access Management (IAM) untuk mencantumkan konten bucket yang ditentukan. Untuk contoh kebijakan bucket yang memberikan akses ke bucket S3, lihat Mengizinkan akses pengguna IAM ke salah satu bucket Anda. Jika Anda mengalami kesalahan HTTP Access Denied (403 Forbidden), lihat Kebijakan bucket dan kebijakan IAM.

Anda dapat membuat daftar bucket dengan menggunakan konsol Amazon S3, the AWS CLI, atau SDK AWS .

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dari daftar Bucket tujuan umum, pilih bucket yang ingin Anda lihat.

> **Note**
>
> Daftar ember tujuan umum mencakup ember yang terletak di semua. Wilayah AWS

Menggunakan AWS CLI

Untuk menggunakan AWS CLI untuk mengakses bucket S3 atau membuat daftar bucket S3, gunakan perintah. `ls` Saat mendaftarkan semua objek di dalam bucket Anda, perhatikan bahwa Anda harus memiliki izin `s3:ListBucket`.

Untuk menggunakan perintah contoh ini, ganti *`DOC-EXAMPLE-BUCKET1`* dengan nama bucket Anda.

```
$ aws s3 ls s3://DOC-EXAMPLE-BUCKET1
```

Perintah contoh berikut ini mencantumkan semua bucket Amazon S3 di akun Anda:

```
$ aws s3 ls
```

Untuk informasi dan contoh selengkapnya, lihat Mendaftarkan bucket dan objek.

Menggunakan AWS SDK

Anda juga dapat mengakses bucket Amazon S3 dengan menggunakan pengoperasian API ListBuckets. Untuk contoh cara menggunakan operasi ini dengan AWS SDK yang berbeda, lihatGunakan `ListBuckets` dengan AWS SDK atau CLI.

Membuat bucket

Untuk mengunggah data Anda ke Amazon S3, Anda harus terlebih dahulu membuat bucket Amazon S3 di salah satu Wilayah AWS. Saat Anda membuat bucket, Anda harus memilih nama bucket dan Wilayah. Anda dapat memilih opsi manajemen penyimpanan lain untuk bucket. Setelah membuat bucket, Anda tidak dapat mengubah nama atau Wilayahnya. Untuk informasi tentang penamaan bucket, lihat Peraturan penamaan bucket.

Akun AWS Yang menciptakan ember memilikinya. Anda dapat mengunggah sejumlah objek ke bucket. Secara default, Anda dapat membuat hingga 100 ember di masing-masing ember Anda Akun AWS. Jika Anda memerlukan lebih banyak bucket, Anda dapat meningkatkan batas bucket akun Anda hingga maksimum 1.000 bucket dengan mengirimkan peningkatan batas layanan. Untuk mempelajari cara mengajukan kenaikan batas bucket, lihat Layanan AWS kuota di AWS Referensi Umum. Anda dapat menyimpan berapa pun jumlah objek dalam sebuah bucket.

Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan baik untuk mengontrol kepemilikan objek yang diunggah ke bucket, maupun untuk menonaktifkan atau mengaktifkan daftar kontrol akses (ACL). Secara default, Kepemilikan Objek diatur ke pengaturan yang diberlakukan pemilik Bucket, dan semua ACL dinonaktifkan. Ketika ACL dinonaktifkan, pemilik bucket memiliki setiap objek di dalam bucket, dan mengelola akses ke data secara eksklusif dengan menggunakan kebijakan.

Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Enkripsi di sisi server dengan Amazon S3 managed keys (SSE-S3) adalah tingkat dasar konfigurasi enkripsi untuk setiap bucket di Amazon S3. Semua objek baru yang diunggah ke bucket S3 secara otomatis dienkripsi dengan SSE-S3 sebagai tingkat dasar pengaturan enkripsi. Jika Anda ingin menggunakan jenis enkripsi default yang berbeda, Anda juga dapat menentukan enkripsi di sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS) atau kunci yang disediakan pelanggan (SSE-C) untuk mengenkripsi data Anda. Untuk informasi selengkapnya, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3.

Anda dapat menggunakan konsol Amazon S3, API Amazon S3 AWS CLI, AWS atau SDK untuk membuat bucket. Untuk informasi selengkapnya tentang izin yang diperlukan untuk membuat bucket, lihat CreateBucketdi Referensi API Amazon Simple Storage Service.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di bilah navigasi di bagian atas halaman, pilih nama yang saat ini ditampilkan Wilayah AWS. Selanjutnya, pilih Wilayah tempat Anda ingin membuat ember.

   > **Note**
   >
   > Untuk meminimalkan latensi dan biaya serta memenuhi persyaratan regulasi, pilih Wilayah yang dekat dengan Anda. Objek yang disimpan di Wilayah tidak pernah keluar dari Wilayah kecuali Anda secara tegas mentransfer atau mereplikasinya ke Wilayah lain. Untuk daftar Amazon S3 Wilayah AWS, lihat Layanan AWS titik akhir di. Referensi Umum Amazon Web Services

3. Di panel navigasi kiri, pilih Bucket.
4. Pilih Buat bucket.

   Halaman Buat bucket terbuka.
5. Di bawah Konfigurasi umum, lihat Wilayah AWS tempat bucket Anda akan dibuat.
6. Di bawah jenis Bucket, pilih Tujuan umum.
7. Untuk Nama bucket, masukkan nama untuk bucket Anda.

   Nama bucket harus:

   - Unik dalam partisi. Partisi adalah pembuatan grup Wilayah. Saat ini, AWS memiliki tiga partisi: `aws` (Wilayah Standar), `aws-cn` (Wilayah Tiongkok), dan `aws-us-gov` (AWS GovCloud (US) Regions).
   - Panjangnya antara 3 hingga 63 karakter.
   - Hanya terdiri dari huruf kecil, angka, titik (.), dan tanda hubung (-). Untuk kompatibilitas terbaik, kami menyarankan agar Anda menghindari penggunaan titik (.) dalam nama bucket, kecuali untuk bucket yang digunakan hanya untuk menghosting situs web statis.
   - Dimulai dan diakhiri dengan huruf atau angka.

Setelah membuat bucket, Anda tidak dapat mengubah namanya. Untuk informasi selengkapnya tentang penamaan bucket, lihat Peraturan penamaan bucket.

> **Important**
>
> Hindari menyertakan informasi sensitif, seperti nomor akun, dalam nama bucket. Nama bucket terlihat dalam URL yang menunjuk objek dalam bucket.

8. AWS Management Console memungkinkan Anda menyalin pengaturan bucket yang ada ke bucket baru Anda. Jika Anda tidak ingin menyalin pengaturan bucket yang ada, lewati ke langkah berikutnya.

> **Note**
>
> Opsi ini:
>
> - Tidak tersedia di AWS CLI dan hanya tersedia di konsol
> - Tidak tersedia untuk bucket direktori
> - Tidak menyalin kebijakan bucket dari bucket yang ada ke bucket baru

   Untuk menyalin setelan bucket yang ada, di bagian Salin setelan dari bucket yang ada, pilih Pilih bucket. Jendela Choose bucket terbuka. Temukan bucket dengan pengaturan yang ingin Anda salin, lalu pilih Pilih bucket. Jendela Choose bucket ditutup, dan jendela Create bucket terbuka kembali.

   Di bawah Salin pengaturan dari bucket yang ada, Anda sekarang akan melihat nama bucket yang Anda pilih. Anda juga akan melihat opsi Restore default yang dapat Anda gunakan untuk menghapus pengaturan bucket yang disalin. Tinjau setelan bucket yang tersisa, di halaman Buat bucket. Anda akan melihat bahwa mereka sekarang cocok dengan pengaturan ember yang Anda pilih. Anda dapat melompat ke langkah terakhir.

9. Di bawah Kepemilikan Objek, untuk menonaktifkan atau mengaktifkan ACL dan mengontrol kepemilikan objek yang diunggah di bucket Anda, pilih salah satu pengaturan berikut ini:

ACL dinonaktifkan

- Pemilik bucket yang diberlakukan (default)–ACL dinonaktifkan, dan pemilik bucket secara otomatis memilikinya, serta mendapatkan kontrol penuh atas setiap objek di dalam bucket. ACL tidak lagi memengaruhi izin akses ke data di bucket S3. Bucket menggunakan kebijakan secara eksklusif untuk menentukan kontrol akses.

  Secara default, ACL dinonaktifkan. Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL. Kami menyarankan agar ACL dinonaktifkan, kecuali dalam keadaan yang tidak biasa ketika Anda perlu mengontrol akses untuk setiap objek secara individual. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

ACL diaktifkan

- Pemilik bucket yang dipilih—Pemilik bucket memiliki dan diberikan kendali penuh atas objek baru yang ditulis akun lain ke bucket dengan ACL `bucket-owner-full-control` yang dibatasi.

  Jika Anda menerapkan pengaturan Pemilik bucket yang dipilih, agar semua unggahan Amazon S3 menyertakan ACL `bucket-owner-full-control` yang terekam, Anda dapat menambahkan kebijakan bucket yang hanya mengizinkan unggahan objek yang menggunakan ACL ini.
- Penulis objek — Akun AWS Yang mengunggah objek memiliki objek, memiliki kontrol penuh atasnya, dan dapat memberikan pengguna lain akses ke sana melalui ACL.

> **Note**
>
> Pengaturan default-nya adalah Pemilik Bucket yang diberlakukan. Untuk menerapkan pengaturan default dan menonaktifkan ACL, hanya izin `s3:CreateBucket` yang diperlukan. Untuk mengaktifkan ACL, Anda harus memiliki izin `s3:PutBucketOwnershipControls`.

10. Di bawah Pengaturan Blokir Akses Publik untuk bucket ini, pilih pengaturan Blokir Akses Publik yang ingin Anda terapkan ke bucket.

Secara default, semua pengaturan Blokir Akses Publik untuk bucket direktori diaktifkan. Kami menyarankan Anda tetap mengaktifkan semua pengaturan, kecuali Anda tahu bahwa Anda perlu menonaktifkan satu atau beberapa pengaturan untuk kasus penggunaan spesifik Anda. Untuk informasi lebih lanjut tentang pemblokiran akses publik, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

> **Note**
>
> Untuk mengaktifkan semua pengaturan Blokir Akses Publik, hanya izin `s3:CreateBucket` yang diperlukan. Untuk mematikan pengaturan Blokir Akses Publik, Anda harus memiliki izin `s3:PutBucketPublicAccessBlock`.

11. (Opsional) Di bawah Penentuan Versi Bucket, Anda dapat memilih apakah Anda ingin menyimpan varian objek di bucket Anda. Untuk informasi selengkapnya tentang penentuan versi, lihat Menggunakan Penentuan Versi dalam bucket S3.

    Untuk menonaktifkan atau mengaktifkan Penentuan Versi di bucket Anda, pilih Nonaktifkan atau Aktifkan.

12. (Opsional) Di bawah Tanda, Anda dapat memilih untuk menambahkan tanda ke bucket Anda. Tanda adalah pasangan kunci-nilai yang digunakan untuk mengategorikan penyimpanan.

    Untuk menambahkan tanda bucket, masukkan Kunci dan secara opsional Nilai, lalu pilih Tambahkan Tanda.

13. Di bagian bawah Enkripsi default, pilih Edit.
14. Untuk mengonfigurasi enkripsi default, di bawah Jenis enkripsi, pilih salah satu dari berikut ini:

    - Kunci yang dikelola Amazon S3 (SSE-S3)
    - AWS Key Management Service kunci (SSE-KMS)

    > **Important**
    >
    > Jika Anda menggunakan opsi SSE-KMS untuk konfigurasi enkripsi default, Anda tunduk pada kuota permintaan per detik (RPS) AWS KMS. Untuk informasi selengkapnya tentang AWS KMS kuota dan cara meminta kenaikan kuota, lihat Kuota di Panduan Pengembang AWS Key Management Service .

Bucket dan objek baru dienkripsi dengan enkripsi di sisi server dengan kunci yang dikelola Amazon S3 sebagai tingkat dasar konfigurasi enkripsi. Untuk informasi selengkapnya tentang enkripsi default, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3.

Untuk informasi selengkapnya tentang penggunaan enkripsi di sisi server Amazon S3 guna mengenkripsi data Anda, lihat Menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3).

15. Jika Anda memilih kunci AWS Key Management Service (SSE-KMS), lakukan hal berikut:

    a. Di bawah AWS KMS kunci, tentukan kunci KMS Anda dengan salah satu cara berikut ini:

    - Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari Anda AWS KMS keys, dan pilih kunci KMS Anda dari daftar kunci yang tersedia.

      Kunci Kunci yang dikelola AWS (`aws/s3`) dan kunci terkelola pelanggan Anda muncul dalam daftar ini. Untuk informasi selengkapnya tentang CMK, lihat Kunci pelanggan dan AWS kunci di AWS Key Management Service Panduan Pengembang.
    - Untuk memasukkan ARN kunci KMS, pilih Masukkan AWS KMS key ARN, dan masukkan ARN kunci KMS Anda di bidang yang muncul.
    - Untuk membuat kunci terkelola pelanggan baru di AWS KMS konsol, pilih Buat kunci KMS.

      Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat kunci di Panduan AWS Key Management Service Pengembang.

    > **Important**
    >
    > Anda hanya dapat menggunakan tombol KMS yang tersedia Wilayah AWS sama dengan bucket. Konsol Amazon S3 hanya mencantumkan kunci 100 KMS pertama di Wilayah yang sama dengan bucket. Untuk menggunakan kunci KMS yang tidak terdaftar, Anda harus memasukkan ARN kunci KMS Anda. Jika Anda ingin menggunakan kunci KMS yang dimiliki oleh akun lain, Anda harus terlebih dahulu memiliki izin untuk menggunakan kunci tersebut, dan kemudian Anda harus memasukkan ARN kunci KMS. Untuk informasi selengkapnya tentang izin lintas akun untuk kunci KMS, lihat Membuat kunci KMS yang dapat digunakan akun lain di Panduan Pengembang AWS Key Management Service . Untuk informasi

selengkapnya tentang SSE-KMS, lihat Menentukan enkripsi di sisi server dengan AWS KMS (SSE-KMS).
Saat Anda menggunakan enkripsi sisi server AWS KMS key untuk Amazon S3, Anda harus memilih kunci KMS enkripsi simetris. Amazon S3 hanya mendukung kunci KMS enkripsi simetris dan tidak mendukung kunci KMS asimetris. Untuk informasi selengkapnya, lihat Mengidentifikasi tombol KMS simetris dan asimetris dalam Panduan Pengembang AWS Key Management Service .

Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat kunci di Panduan AWS Key Management Service Pengembang. Untuk informasi selengkapnya tentang penggunaan AWS KMS dengan Amazon S3, lihat. Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS)

b. Saat mengonfigurasi bucket untuk menggunakan enkripsi default dengan SSE-KMS, Anda juga dapat mengaktifkan Kunci Bucket S3. S3 Bucket Keys menurunkan biaya enkripsi dengan mengurangi lalu lintas permintaan dari Amazon S3 ke. AWS KMS Untuk informasi selengkapnya, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3.

   Untuk menggunakan Kunci Bucket S3, di bagian bawah Kunci Bucket, pilih Aktifkan.

16. (Opsional) Jika Anda ingin mengaktifkan Kunci Objek S3, lakukan hal berikut ini:

    a. Pilih Pengaturan lanjutan.

    > **Important**
    >
    > Mengaktifkan Kunci Objek juga mengaktifkan Penentuan Versi untuk bucket. Setelah mengaktifkan, Anda harus mengonfigurasi pengaturan penyimpanan default Kunci Objek dan penahanan legal untuk melindungi objek baru agar tidak dihapus atau ditimpa.

    b. Jika Anda ingin mengaktifkan Kunci Objek, pilih Aktifkan, baca peringatan yang muncul, lalu setujui.

Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek S3.

> **Note**
>
> Untuk membuat bucket dengan dukungan Kunci Objek, Anda harus memiliki izin berikut: `s3:CreateBucket`, `s3:PutBucketVersioning` dan `s3:PutBucketObjectLockConfiguration`.

17. Pilih Buat bucket.

Menggunakan AWS SDK

Saat Anda menggunakan AWS SDK untuk membuat bucket, Anda harus membuat klien dan kemudian menggunakan klien untuk mengirim permintaan untuk membuat bucket. Sebagai praktik terbaik, Anda harus membuat klien dan bucket di Wilayah AWS yang sama. Jika Anda tidak menentukan Wilayah saat membuat klien atau bucket, Amazon S3 menggunakan Wilayah default AS Timur (Virginia Utara). Jika Anda ingin membatasi pembuatan bucket ke Wilayah AWS tertentu, gunakan kunci kondisi `LocationConstraint`.

Untuk membuat klien agar dapat mengakses titik akhir dual-stack, Anda harus menentukan Wilayah AWS. Untuk informasi selengkapnya, lihat Titik akhir tumpukan ganda. Untuk daftar yang tersedia Wilayah AWS, lihat Wilayah dan titik akhir di. Referensi Umum AWS

Saat Anda membuat klien, Wilayah akan memetakan ke titik akhir spesifik Wilayah. Klien menggunakan titik akhir ini untuk berkomunikasi dengan Amazon S3: `s3.`*`region`*`.amazonaws.com`. Jika Wilayah Anda diluncurkan setelah 20 Maret 2019, klien dan bucket Anda harus berada di dalam Wilayah yang sama. Namun, Anda dapat menggunakan klien di Wilayah AS Timur (Virginia Utara) untuk membuat bucket di setiap Wilayah yang diluncurkan sebelum 20 Maret 2019. Untuk informasi selengkapnya, lihat Titik akhir warisan.

Contoh kode AWS SDK ini melakukan tugas-tugas berikut:

- Membuat klien dengan secara eksplisit menentukan Wilayah AWS–Dalam contoh, klien menggunakan titik akhir `s3.us-west-2.amazonaws.com` untuk berkomunikasi dengan Amazon S3. Anda dapat menentukan Wilayah AWS. Untuk daftar Wilayah AWS, lihat Wilayah dan titik akhir dalam Referensi AWS Umum.
- Kirim permintaan pembuatan bucket dengan menentukan hanya nama bucket–Klien mengirimkan permintaan ke Amazon S3 untuk membuat bucket di Wilayah tempat Anda membuat klien.

- Ambil informasi tentang lokasi bucket–Amazon S3 menyimpan informasi lokasi bucket di sub-sumber daya lokasi yang dikaitkan dengan bucket.

Java

Contoh ini menunjukkan cara untuk membuat bucket Amazon S3 menggunakan AWS SDK for Java. Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CreateBucketRequest;
import com.amazonaws.services.s3.model.GetBucketLocationRequest;

import java.io.IOException;

public class CreateBucket2 {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            if (!s3Client.doesBucketExistV2(bucketName)) {
                // Because the CreateBucketRequest object doesn't specify a region,
 the
                // bucket is created in the region specified in the client.
                s3Client.createBucket(new CreateBucketRequest(bucketName));

                // Verify that the bucket was created by retrieving it and checking
 its
                // location.
```

```
                String bucketLocation = s3Client.getBucketLocation(new
 GetBucketLocationRequest(bucketName));
                System.out.println("Bucket location: " + bucketLocation);
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Untuk informasi tentang cara membuat dan menguji sampel yang berfungsi, lihat AWS SDK for .NET Versi 3 API Referensi.

Example

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using Amazon.S3.Util;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CreateBucketTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            CreateBucketAsync().Wait();
```

```
        }

        static async Task CreateBucketAsync()
        {
            try
            {
                if (!(await AmazonS3Util.DoesS3BucketExistAsync(s3Client,
bucketName)))
                {
                    var putBucketRequest = new PutBucketRequest
                    {
                        BucketName = bucketName,
                        UseClientRegion = true
                    };

                    PutBucketResponse putBucketResponse = await
s3Client.PutBucketAsync(putBucketRequest);
                }
                // Retrieve the bucket location.
                string bucketLocation = await FindBucketLocationAsync(s3Client);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
        }
        static async Task<string> FindBucketLocationAsync(IAmazonS3 client)
        {
            string bucketLocation;
            var request = new GetBucketLocationRequest()
            {
                BucketName = bucketName
            };
            GetBucketLocationResponse response = await
client.GetBucketLocationAsync(request);
            bucketLocation = response.Location.ToString();
            return bucketLocation;
        }
```

```
    }
}
```

Ruby

Untuk informasi tentang cara membuat dan menguji sampel kerja, lihat AWS SDK for Ruby - Versi 3.

Example

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket actions.
class BucketCreateWrapper
  attr_reader :bucket

  # @param bucket [Aws::S3::Bucket] An Amazon S3 bucket initialized with a name.
 This is a client-side object until
  #                                 create is called.
  def initialize(bucket)
    @bucket = bucket
  end

  # Creates an Amazon S3 bucket in the specified AWS Region.
  #
  # @param region [String] The Region where the bucket is created.
  # @return [Boolean] True when the bucket is created; otherwise, false.
  def create?(region)
    @bucket.create(create_bucket_configuration: { location_constraint: region })
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't create bucket. Here's why: #{e.message}"
    false
  end

  # Gets the Region where the bucket is located.
  #
  # @return [String] The location of the bucket.
  def location
    if @bucket.nil?
      "None. You must create a bucket before you can get its location!"
    else
      @bucket.client.get_bucket_location(bucket: @bucket.name).location_constraint
```

```
    end
  rescue Aws::Errors::ServiceError => e
    "Couldn't get the location of #{@bucket.name}. Here's why: #{e.message}"
  end
end

# Example usage:
def run_demo
  region = "us-west-2"
  wrapper = BucketCreateWrapper.new(Aws::S3::Bucket.new("doc-example-bucket-
#{Random.uuid}"))
  return unless wrapper.create?(region)

  puts "Created bucket #{wrapper.bucket.name}."
  puts "Your bucket's region is: #{wrapper.location}"
end

run_demo if $PROGRAM_NAME == __FILE__
```

Menggunakan AWS CLI

Anda juga dapat menggunakan AWS Command Line Interface (AWS CLI) untuk membuat bucket S3. Untuk informasi selengkapnya, lihat create-bucket dalam AWS CLI Referensi Perintah.

Untuk informasi tentang AWS CLI, lihat Apa itu AWS Command Line Interface? dalam AWS Command Line Interface User Guide.

Melihat properti untuk bucket S3

Anda dapat melihat properti untuk bucket Amazon S3 apa pun yang Anda miliki. Pengaturan ini meliputi:

- Penentuan Versi Bucket–Menyimpan beberapa versi objek dalam satu bucket dengan menggunakan Penentuan Versi. Secara default, Penentuan Versi dinonaktifkan untuk bucket yang baru. Untuk informasi tentang Penentuan Versi, lihat Mengaktifkan Penentuan Versi pada bucket.
- Tag — Dengan alokasi AWS biaya, Anda dapat menggunakan tag bucket untuk membubuhi keterangan penagihan penggunaan bucket. Tanda adalah pasangan nilai kunci yang mewakili label yang Anda tetapkan ke bucket. Untuk informasi selengkapnya, lihat Menggunakan tag alokasi biaya bucket S3.

- Enkripsi default–Mengaktifkan enkripsi default memberikan Anda enkripsi di sisi server otomatis. Amazon S3 mengenkripsi sebuah objek sebelum menyimpannya ke disk, dan mendekripsi objek tersebut saat Anda mengunduhnya. Untuk informasi selengkapnya, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3.
- Pencatatan log akses server–Menyediakan catatan terperinci untuk permintaan yang dilakukan ke bucket Anda dengan mencatat log akses server. Secara default, Amazon S3 tidak mengumpulkan log akses server. Untuk informasi tentang mengaktifkan pencatatan akses server, lihatMengaktifkan pencatatan akses server Amazon S3.
- AWS CloudTrail peristiwa data - Gunakan CloudTrail untuk mencatat peristiwa data. Secara default, jejak tidak mencatat peristiwa data. Biaya tambahan berlaku untuk peristiwa data. Untuk informasi lebih lanjut, lihat Peristiwa Pencatatan Data untuk Pelacakan dalam AWS CloudTrail Panduan Pengguna.
- Pemberituhan peristiwa–Mengaktifkan peristiwa bucket Amazon S3 tertentu untuk mengirim pesan pemberitahuan ke destinasi, kapan pun peristiwa tersebut terjadi. Untuk informasi selengkapnya, lihat Mengaktifkan dan mengonfigurasi notifikasi peristiwa menggunakan konsol Amazon S3.
- Transfer acceleration–Memungkinkan pentransferan file yang cepat, mudah, dan aman melalui jarak jauh antara klien Anda dan bucket S3. Untuk informasi lebih lanjut tentang Transfer Acceleration, lihat Mengaktifkan dan menggunakan S3 Transfer Acceleration.
- Kunci Objek–Gunakan Kunci Objek S3 untuk mencegah objek terhapus atau ditimpa selama jangka waktu tertentu, atau tanpa batas waktu. Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek S3.
- Pembayaran Pemohon–Mengaktifkan Pembayaran Pemohon jika Anda menginginkan pemohon (bukan pemilik bucket) untuk membayar permintaan dan transfer data. Untuk informasi selengkapnya, lihat Menggunakan bucket Pembayaran Pemohon untuk transfer dan penggunaan penyimpanan.
- Hosting situs web statis–Anda dapat menghosting situs web statis di Amazon S3. Untuk informasi selengkapnya, lihat Hosting situs web statis menggunakan Amazon S3.

Anda dapat melihat properti bucket menggunakan AWS Management Console, AWS CLI, atau AWS SDK

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di dalam daftar Bucket, pilih nama bucket yang ingin Anda lihat propertinya.
3. Pilih tab Properti.
4. Pada halaman Properties, Anda dapat mengonfigurasi properti di atas untuk bucket.

Menggunakan AWS CLI

Lihat properti bucket dengan AWS CLI

Perintah berikut menunjukkan bagaimana Anda dapat menggunakan AWS CLI untuk membuat daftar properti bucket yang berbeda.

Berikut ini mengembalikan set tag yang terkait dengan bucket *example-s3-bucket1*. Untuk informasi selengkapnya tentang tag bucket lihat,Menggunakan tag alokasi biaya bucket S3.

```
aws s3api get-bucket-tagging --bucket example-s3-bucket1
```

Untuk informasi dan contoh selengkapnya, lihat get-bucket-tagging di AWS CLI Referensi Perintah.

Berikut ini mengembalikan status pembuatan versi bucket example-s3-bucket1. Untuk informasi tentang pembuatan versi bucket, lihat. Menggunakan Penentuan Versi dalam bucket S3

```
aws s3api get-bucket-versioning --bucket example-s3-bucket1
```

Untuk informasi dan contoh selengkapnya, lihat get-bucket-versioning di AWS CLI Referensi Perintah.

Berikut ini mengembalikan konfigurasi enkripsi default untuk bucket *example-s3-bucket1*. Secara default, semua bucket memiliki konfigurasi enkripsi default yang menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3). Untuk informasi tentang enkripsi default bucket, lihatMengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3.

```
aws s3api get-bucket-encryption --bucket example-s3-bucket1
```

Untuk informasi dan contoh selengkapnya, lihat get-bucket-encryption di AWS CLI Referensi Perintah.

Berikut ini mengembalikan konfigurasi notifikasi bucket *example-s3-bucket1*. Untuk informasi tentang notifikasi acara bucket, lihatNotifikasi Peristiwa Amazon S3.

```
aws s3api get-bucket-notification-configuration --bucket example-s3-bucket1
```

Untuk informasi dan contoh selengkapnya, lihat get-bucket-notification-configuration di AWS CLI Referensi Perintah.

Berikut ini mengembalikan status logging untuk bucket `example-s3-bucket1`. Untuk informasi tentang pencatatan bucket, lihatPencatatan permintaan dengan pencatatan akses server.

```
aws s3api get-bucket-logging --bucket example-s3-bucket1
```

Untuk informasi dan contoh selengkapnya, lihat get-bucket-logging di AWS CLI Referensi Perintah.

Menggunakan AWS SDK

Untuk contoh cara mengembalikan properti bucket dengan AWS SDK, seperti pembuatan versi, tag, dan lainnya, lihat. Tindakan untuk Amazon S3 menggunakan SDK AWS

Untuk informasi umum tentang penggunaan AWS SDK yang berbeda, lihatMengembangkan dengan Amazon S3 menggunakan SDK AWS.

Mengosongkan bucket

Anda dapat mengosongkan konten bucket menggunakan konsol Amazon S3, AWS SDK, atau AWS Command Line Interface ().AWS CLI Saat mengosongkan bucket direktori, Anda menghapus semua objeknya, namun tetap menyimpan bucket tersebut. Setelah Anda mengosongkan bucket, tindakan ini tidak dapat dibatalkan. Objek yang ditambahkan ke bucket saat tindakan mengosongkan bucket sedang berlangsung akan dihapus. Semua objek (termasuk semua versi objek dan penanda hapus) di dalam bucket harus dihapus sebelum bucket itu sendiri dapat dihapus.

Saat Anda mengosongkan bucket yang memiliki Penentuan Versi S3 yang diaktifkan atau ditangguhkan, semua versi dari semua objek dalam bucket tersebut akan dihapus. Untuk informasi selengkapnya, lihat Bekerja dengan objek di dalam bucket dengan dukungan Penentuan Versi.

Anda juga dapat menentukan konfigurasi siklus hidup pada bucket untuk objek yang kedaluwarsa, sehingga Amazon S3 dapat menghapusnya. Untuk informasi selengkapnya, lihat Menyetel konfigurasi siklus hidup pada bucket. Untuk mengosongkan bucket berukuran besar, sebaiknya gunakan aturan konfigurasi Siklus Hidup S3. Kedaluwarsa siklus hidup merupakan proses asinkron, sehingga aturannya mungkin memerlukan waktu beberapa hari untuk dijalankan sebelum bucket

tersebut kosong. Setelah pertama kali Amazon S3 menjalankan aturan, semua objek yang memenuhi syarat untuk kedaluwarsa ditandai untuk dihapus. Anda tidak lagi dikenakan biaya untuk objek yang ditandai untuk dihapus. Untuk informasi selengkapnya, lihat Bagaimana saya mengosongkan bucket Amazon S3 menggunakan aturan konfigurasi siklus hidup?.

Menggunakan konsol S3

Anda dapat menggunakan konsol Amazon S3 untuk mengosongkan bucket, yang menghapus semua objek di dalam bucket tanpa menghapus bucket itu sendiri.

Untuk mengosongkan bucket S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Dalam daftar Nama bucket, pilih opsi di samping nama bucket yang ingin Anda kosongkan, lalu pilih Kosongkan.
3. Pada halaman Bucket kosong, konfirmasikan bahwa Anda ingin mengosongkan bucket dengan memasukkan nama bucket ke dalam bidang teks, lalu pilih Kosongkan.
4. Pantau progres proses pengosongan bucket di halaman Bucket kosong: Status.

Menggunakan AWS CLI

Anda dapat mengosongkan bucket AWS CLI hanya menggunakan bucket jika bucket tidak mengaktifkan Bucket Versioning. Jika pembuatan versi tidak diaktifkan, Anda dapat menggunakan AWS CLI perintah `rm` (hapus) dengan `--recursive` parameter untuk mengosongkan bucket (atau menghapus subset objek dengan awalan nama kunci tertentu).

Perintah `rm` berikut menghapus objek yang memiliki prefiks nama kunci `doc`, misalnya, `doc/doc1` dan `doc/doc2`.

```
$ aws s3 rm s3://bucket-name/doc --recursive
```

Gunakan perintah berikut untuk menghapus semua objek tanpa menentukan prefiks.

```
$ aws s3 rm s3://bucket-name --recursive
```

Untuk informasi lebih lanjut, lihat Menggunakan perintah S3 tingkat tinggi dengan AWS CLI di AWS Command Line Interface Panduan Pengguna.

> **Note**
>
> Anda tidak dapat menghapus objek dari bucket dengan Penentuan Versi diaktifkan. Amazon S3 menambahkan penanda hapus saat Anda menghapus objek, yang dilakukan oleh perintah ini. Untuk informasi selengkapnya tentang Penentuan Versi Bucket S3, lihat Menggunakan Penentuan Versi dalam bucket S3.

Menggunakan AWS SDK

Anda dapat menggunakan AWS SDK untuk mengosongkan bucket atau menghapus subset objek yang memiliki awalan nama kunci tertentu.

Untuk contoh cara mengosongkan ember menggunakan AWS SDK for Java, lihatMenghapus bucket. Kode tersebut menghapus semua objek, terlepas dari apakah bucket telah mengaktifkan Penentuan Versi atau tidak, lalu menghapus bucket. Untuk mengosongkan bucket saja, pastikan bahwa Anda menghapus pernyataan yang menghapus bucket.

Untuk informasi selengkapnya tentang menggunakan AWS SDK lain, lihat Alat untuk Amazon Web Services.

Menggunakan konfigurasi siklus aktif

Untuk mengosongkan bucket berukuran besar, sebaiknya gunakan aturan konfigurasi Siklus Hidup S3. Kedaluwarsa siklus hidup merupakan proses asinkron, sehingga aturannya mungkin memerlukan waktu beberapa hari untuk dijalankan sebelum bucket tersebut kosong. Setelah pertama kali Amazon S3 menjalankan aturan, semua objek yang memenuhi syarat untuk kedaluwarsa ditandai untuk dihapus. Anda tidak lagi dikenakan biaya untuk objek yang ditandai untuk dihapus. Untuk informasi selengkapnya, lihat Bagaimana saya mengosongkan bucket Amazon S3 menggunakan aturan konfigurasi siklus hidup?.

Jika Anda menggunakan konfigurasi siklus hidup untuk mengosongkan bucket, konfigurasi harus mencakup versi terkini, versi terdahulu, penanda hapus, dan unggahan multibagian yang tidak lengkap.

Anda dapat menambahkan aturan konfigurasi siklus hidup untuk menjadikan semua objek atau subset objek yang memiliki prefiks nama kunci tertentu menjadi kedaluwarsa. Misalnya, untuk menghapus semua objek dalam bucket, Anda dapat menyetel aturan siklus hidup untuk kedaluwarsa objek satu hari setelah pembuatannya.

Amazon S3 mendukung aturan siklus hidup bucket yang dapat Anda gunakan untuk menghentikan unggahan multibagian yang tidak selesai dalam jumlah hari tertentu setelah dimulai. Kami menyarankan Anda untuk mengonfigurasi aturan siklus hidup ini, guna meminimalkan biaya penyimpanan Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi siklus hidup bucket untuk menghapus unggahan multibagian yang tidak lengkap.

Untuk informasi lebih lanjut tentang penggunaan konfigurasi siklus hidup untuk mengosongkan bucket, lihat Menyetel konfigurasi siklus hidup pada bucket dan Mengakhiri objek.

Mengosongkan ember dengan dikonfigurasi AWS CloudTrail

AWS CloudTrail melacak peristiwa data tingkat objek di bucket Amazon S3, seperti menghapus objek. Jika Anda menggunakan bucket sebagai tujuan untuk mencatat CloudTrail peristiwa dan menghapus objek dari bucket yang sama, Anda mungkin membuat objek baru sambil mengosongkan bucket Anda. Untuk mencegah hal ini, hentikan AWS CloudTrail jejak Anda. Untuk informasi selengkapnya tentang menghentikan CloudTrail jejak Anda dari peristiwa pencatatan, lihat Menonaktifkan pencatatan untuk jejak di Panduan AWS CloudTrail Pengguna.

Alternatif lain untuk menghentikan CloudTrail jejak agar tidak ditambahkan ke bucket adalah dengan menambahkan `s3:PutObject` pernyataan penolakan ke kebijakan bucket Anda. Jika Anda ingin menyimpan objek baru di bucket di lain waktu Anda harus menghapus pernyataan penolakan `s3:PutObject` ini. Untuk informasi selengkapnya, lihat Operasi objek dan Elemen kebijakan JSON IAM: Efek dalam Panduan Pengguna IAM.

Menghapus bucket

Anda dapat menghapus bucket Amazon S3 yang kosong. Sebelum menghapus bucket, pertimbangkan hal berikut ini:

- Nama bucket unik. Jika Anda menghapus bucket, AWS pengguna lain dapat menggunakan nama tersebut.
- Jika bucket menghosting situs web statis, dan Anda membuat serta mengonfigurasi zona Amazon Route 53 yang di-hosting seperti yang dijelaskan dalam Tutorial: Mengonfigurasi situs web statis menggunakan domain kustom yang terdaftar di Route 53, Anda harus membersihkan pengaturan zona Route 53 yang di-hosting yang berkaitan dengan bucket tersebut. Untuk informasi selengkapnya, lihat Langkah 2: Menghapus Route 53 zona yang di-hosting.
- Jika bucket menerima data log dari Elastic Load Balancing (ELB): Kami sarankan agar Anda menghentikan pengiriman log ELB ke bucket sebelum menghapusnya. Setelah Anda menghapus

bucket, jika pengguna lain membuat bucket menggunakan nama yang sama, data log Anda berpotensi dikirim ke bucket tersebut. Untuk informasi tentang log akses ELB, lihat Mengakses log dalam Panduan Pengguna untuk Penyeimbang Beban Klasik dan Mengakses log dalam Panduan Pengguna untuk Penyeimbang Beban Aplikasi.

Pemecahan Masalah

Jika Anda tidak dapat menghapus bucket Amazon S3, pertimbangkan hal berikut ini:

- Pastikan bucket kosong–Anda hanya dapat menghapus bucket yang tidak memiliki objek di dalamnya. Pastikan bucket kosong.
- Pastikan tidak ada titik akses yang terpasang–Anda hanya dapat menghapus bucket yang tidak memiliki titik akses yang melekat padanya. Hapus titik akses apa pun yang dilampirkan ke bucket, sebelum menghapus bucket tersebut.
- AWS Organizations kebijakan kontrol layanan (SCP) — Kebijakan kontrol layanan dapat menolak izin penghapusan pada bucket. Untuk informasi selengkapnya tentang SCP, lihat Kebijakan kontrol layanan di AWS Organizations Panduan Pengguna.
- s3: DeleteBucket izin — Jika Anda tidak dapat menghapus bucket, bekerjalah dengan administrator IAM Anda untuk mengonfirmasi bahwa Anda memiliki izin. `s3:DeleteBucket` Untuk informasi tentang cara melihat atau memperbarui izin IAM, lihat Mengubah izin untuk pengguna IAM di Panduan Pengguna IAM.
- s3: DeleteBucket deny statement — Jika Anda memiliki `s3:DeleteBucket` izin dalam kebijakan IAM Anda dan Anda tidak dapat menghapus bucket, kebijakan bucket mungkin menyertakan pernyataan penolakan untuk. `s3:DeleteBucket` Bucket yang dibuat oleh ElasticBeanstalk memiliki kebijakan yang berisi pernyataan ini secara default. Sebelum Anda dapat menghapus bucket, Anda harus menghapus pernyataan ini, atau kebijakan bucket tersebut.

> **Important**
>
> Nama bucket bersifat unik. Jika Anda menghapus bucket, AWS pengguna lain dapat menggunakan nama tersebut. Jika Anda ingin terus menggunakan nama bucket yang sama, jangan menghapus bucket. Kami menyarankan agar Anda mengosongkan bucket dan menyimpannya.

Menggunakan konsol S3

Untuk menghapus S3 bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di Bucket, pilih opsi di samping nama bucket yang ingin Anda hapus, lalu pilih Hapus di bagian atas halaman.
3. Di halaman Hapus bucket, konfirmasikan bahwa Anda ingin menghapus bucket dengan memasukkan nama bucket ke dalam bidang teks, lalu pilih Hapus bucket.

> **Note**
>
> Jika bucket berisi objek apa pun, kosongkan bucket sebelum menghapusnya dengan memilih tautan konfigurasi bucket kosong di peringatan kesalahan Bucket ini tidak kosong dan mengikuti instruksi di halaman Kosongkan bucket. Lalu kembali ke halaman Hapus bucket dan hapus bucket tersebut.

4. Untuk memverifikasi bahwa Anda telah menghapus bucket, buka daftar Bucket dan masukkan nama bucket yang Anda hapus. Jika bucket tidak dapat ditemukan, penghapusan Anda berhasil.

Menggunakan AWS SDK for Java

Contoh berikut menunjukkan cara menghapus bucket menggunakan AWS SDK for Java. Pertama, kode menghapus objek di dalam bucket, kemudian menghapus bucket. Untuk informasi tentang SDK AWS lainnya, lihat Alat untuk Amazon Web Services.

Java

Contoh Java berikut ini menghapus bucket yang berisi objek. Contoh ini menghapus semua objek, lalu menghapus bucket. Contoh ini berfungsi untuk bucket dengan atau tanpa mengaktifkan Penentuan Versi.

> **Note**
>
> Untuk bucket tanpa Penentuan Versi diaktifkan, Anda dapat menghapus semua objek secara langsung dan kemudian menghapus bucket. Untuk bucket dengan Penentuan Versi diaktifkan, Anda harus menghapus semua versi objek sebelum menghapus bucket.

Untuk petunjuk tentang membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.Iterator;

public class DeleteBucket2 {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Delete all objects from the bucket. This is sufficient
            // for unversioned buckets. For versioned buckets, when you attempt to
 delete
            // objects, Amazon S3 inserts
            // delete markers for all objects, but doesn't delete the object
 versions.
            // To delete objects from versioned buckets, delete all of the object
 versions
            // before deleting
```

```
            // the bucket (see below for an example).
            ObjectListing objectListing = s3Client.listObjects(bucketName);
            while (true) {
                Iterator<S3ObjectSummary> objIter =
objectListing.getObjectSummaries().iterator();
                while (objIter.hasNext()) {
                    s3Client.deleteObject(bucketName, objIter.next().getKey());
                }

                // If the bucket contains many objects, the listObjects() call
                // might not return all of the objects in the first listing. Check
to
                // see whether the listing was truncated. If so, retrieve the next
page of
                // objects
                // and delete them.
                if (objectListing.isTruncated()) {
                    objectListing = s3Client.listNextBatchOfObjects(objectListing);
                } else {
                    break;
                }
            }

            // Delete all object versions (required for versioned buckets).
            VersionListing versionList = s3Client.listVersions(new
ListVersionsRequest().withBucketName(bucketName));
            while (true) {
                Iterator<S3VersionSummary> versionIter =
versionList.getVersionSummaries().iterator();
                while (versionIter.hasNext()) {
                    S3VersionSummary vs = versionIter.next();
                    s3Client.deleteVersion(bucketName, vs.getKey(),
vs.getVersionId());
                }

                if (versionList.isTruncated()) {
                    versionList = s3Client.listNextBatchOfVersions(versionList);
                } else {
                    break;
                }
            }

            // After all objects and object versions are deleted, delete the bucket.
            s3Client.deleteBucket(bucketName);
```

```
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
 couldn't
            // parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Menggunakan AWS CLI

Anda dapat menghapus bucket yang berisi objek dengan AWS CLI jika tidak mengaktifkan versi. Saat Anda menghapus bucket yang berisi objek, semua objek dalam bucket dihapus secara permanen, termasuk objek yang ditransisikan ke kelas penyimpanan `S3 Glacier`.

Jika bucket Anda tidak mengaktifkan versi, Anda dapat menggunakan AWS CLI perintah `rb` (hapus bucket) dengan `--force` parameter untuk menghapus bucket dan semua objek di dalamnya. Perintah ini menghapus semua objek terlebih dahulu, kemudian menghapus bucket.

Jika Penentuan Versi diaktifkan, objek berversi tidak akan dihapus dalam proses ini, yang akan menyebabkan penghapusan bucket gagal karena bucket tidak akan kosong. Untuk informasi selengkapnya tentang menghapus objek berversi, lihat Menghapus versi objek.

```
$ aws s3 rb s3://bucket-name --force
```

Untuk informasi selengkapnya, lihat Menggunakan Perintah S3 Tingkat Tinggi dengan AWS Command Line Interface Panduan Pengguna. AWS Command Line Interface

Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Semua bucket Amazon S3 memiliki enkripsi yang dikonfigurasi secara default, dan objek secara otomatis dienkripsi dengan menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3). Pengaturan enkripsi ini berlaku untuk semua objek di dalam bucket Amazon S3 Anda.

Jika Anda memerlukan kontrol lebih besar atas kunci Anda, seperti mengelola rotasi kunci dan hibah kebijakan akses, Anda dapat memilih untuk menggunakan enkripsi sisi server dengan () kunci AWS Key Management Service (SSE-KMS AWS KMS), atau enkripsi sisi server dua lapis dengan kunci (DSSE-KMS). AWS KMS Untuk informasi selengkapnya tentang mengedit kunci KMS, lihat Mengedit kunci dalam Panduan Pengembang AWS Key Management Service .

> **Note**
>
> Kami telah mengubah bucket untuk mengenkripsi unggahan objek baru secara otomatis. Jika sebelumnya Anda membuat bucket tanpa enkripsi default, Amazon S3 akan mengaktifkan enkripsi secara default untuk bucket menggunakan SSE-S3. Tidak akan ada perubahan pada konfigurasi enkripsi default untuk bucket yang sudah ada yang sudah memiliki konfigurasi SSE-S3 atau SSE-KMS. Jika Anda ingin mengenkripsi objek Anda dengan SSE-KMS, Anda harus mengubah jenis enkripsi di pengaturan bucket Anda. Untuk informasi selengkapnya, lihat Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS).

Saat mengonfigurasi bucket untuk menggunakan enkripsi default dengan SSE-KMS, Anda juga dapat mengaktifkan Kunci Bucket S3 untuk mengurangi lalu lintas permintaan dari Amazon S3 ke AWS KMS dan mengurangi biaya enkripsi. Untuk informasi selengkapnya, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3.

Untuk mengidentifikasi bucket yang mengaktifkan SSE-KMS untuk enkripsi default, Anda dapat menggunakan metrik Lensa Penyimpanan Amazon S3. Lensa Penyimpanan S3 adalah fitur analisis penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan objek. Untuk informasi selengkapnya, lihat Menggunakan Lensa Penyimpanan S3 untuk melindungi data Anda.

Saat Anda menggunakan enkripsi di sisi server, Amazon S3 mengenkripsi objek sebelum menyimpannya ke disk, dan mendekripsinya saat Anda mengunduh objek. Untuk informasi lebih lanjut tentang cara untuk melindungi data menggunakan enkripsi di sisi server dan manajemen kunci enkripsi, lihat Melindungi data dengan enkripsi di sisi klien.

Untuk informasi lebih lanjut tentang izin yang diperlukan untuk enkripsi default, lihat PutBucketEncryption dalam Referensi API Amazon Simple Storage Service.

Anda dapat mengonfigurasi perilaku enkripsi default Amazon S3 untuk bucket S3 dengan menggunakan konsol Amazon S3, SDK, Amazon S3 REST API AWS , dan Command Line Interface (). AWS AWS CLI

Mengenkripsi objek yang ada

Untuk mengenkripsi objek Amazon S3 yang tidak terenkripsi, Anda dapat menggunakan Operasi Batch Amazon S3. Anda harus menyediakan Operasi Batch S3 dengan daftar objek untuk dioperasikan, dan Operasi Batch akan memanggil masing-masing API untuk melakukan operasi tertentu. Anda juga dapat menggunakan Operasi Salin Operasi Batch untuk menyalin objek tidak terenkripsi yang ada, dan menuliskannya kembali ke bucket yang sama sebagai objek terenkripsi. Satu tugas Operasi Batch dapat melakukan operasi tertentu pada miliaran objek yang berisi data sebesar eksabita. Untuk informasi selengkapnya, lihat Melakukan operasi batch berskala besar pada objek Amazon S3 dan postingan AWS Blog Penyimpanan Mengenkripsi objek dengan Operasi Batch Amazon S3.

Anda juga dapat mengenkripsi objek yang ada dengan menggunakan operasi `CopyObject` API atau `copy-object` AWS CLI perintah. Untuk informasi selengkapnya, lihat postingan AWS Blog Penyimpanan Mengenkripsi objek Amazon S3 yang sudah ada dengan AWS CLI.

> **Note**
>
> Bucket Amazon S3 dengan enkripsi bucket default yang diatur ke SSE-KMS tidak dapat digunakan sebagai bucket tujuan the section called "Pencatatan akses server". Hanya enkripsi default SSE-S3 yang didukung untuk bucket tujuan log akses server.

Menggunakan enkripsi SSE-KMS untuk operasi lintas akun

Saat menggunakan enkripsi untuk operasi lintas akun, perhatikan hal berikut:

- Jika Nama Sumber Daya AWS KMS key Amazon (ARN) atau alias tidak disediakan pada waktu permintaan atau melalui konfigurasi enkripsi default bucket, Kunci yang dikelola AWS (`aws/s3`) akan digunakan.
- Jika Anda mengunggah atau mengakses objek S3 dengan menggunakan prinsipal AWS Identity and Access Management (IAM) yang sama Akun AWS dengan kunci KMS Anda, Anda dapat menggunakan (). Kunci yang dikelola AWS `aws/s3`
- Jika Anda ingin memberikan akses lintas akun ke objek S3 Anda, gunakan CMK. Anda dapat mengonfigurasi kebijakan CMK untuk mengizinkan akses dari akun lain.
- Jika Anda menentukan kunci KMS yang dikelola pelanggan, kami sarankan untuk menggunakan ARN kunci KMS yang memenuhi syarat sepenuhnya. Jika Anda menggunakan alias kunci KMS sebagai gantinya, AWS KMS selesaikan kunci dalam akun pemohon. Perilaku ini dapat menghasilkan data yang dienkripsi dengan kunci KMS milik pemohon, dan bukan pemilik bucket.
- Anda harus menentukan kunci yang kepada Anda (pemohon) telah diberikan izin `Encrypt`. Untuk informasi selengkapnya, lihat Mengizinkan pengguna di akun lain untuk menggunakan kunci KMS untuk operasi kriptografi di Panduan Pengembang AWS Key Management Service .

Untuk informasi selengkapnya tentang kapan menggunakan kunci terkelola pelanggan dan kunci KMS AWS terkelola, lihat Haruskah saya menggunakan Kunci yang dikelola AWS atau kunci yang dikelola pelanggan untuk mengenkripsi objek saya di Amazon S3?

Menggunakan enkripsi default menggunakan replikasi

Saat Anda mengaktifkan enkripsi default untuk bucket tujuan replikasi, perilaku enkripsi berikut berlaku:

- Jika objek dalam bucket sumber tidak dienkripsi, objek replika dalam bucket tujuan akan dienkripsi menggunakan pengaturan enkripsi default dari bucket tujuan. Akibatnya, tag entitas (ETag) dari objek sumber berbeda dari ETag objek replika. Jika Anda memiliki aplikasi yang menggunakan ETag, Anda harus memperbarui aplikasi tersebut untuk memperhitungkan perbedaan ini.
- Jika objek dalam bucket sumber dienkripsi menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3), enkripsi sisi server dengan kunci () (SSE-KMS), atau enkripsi sisi server dua lapis dengan AWS Key Management Service kunci (DSSE-KMS), objek replika di bucket tujuan menggunakan jenis enkripsi yang sama dengan AWS KMS objek sumber.AWS KMS Pengaturan enkripsi default bucket tujuan tidak digunakan.

Untuk informasi lebih lanjut tentang penggunaan enkripsi default dengan SSE-KMS, lihat Mereplikasi objek terenkripsi.

Menggunakan Kunci Bucket Amazon S3 dengan enkripsi default

Saat mengonfigurasi bucket untuk menggunakan enkripsi default dengan SSE-KMS, Anda juga dapat mengaktifkan Kunci Bucket S3. S3 Bucket Keys mengurangi jumlah transaksi dari Amazon S3 AWS KMS untuk mengurangi biaya SSE-KMS.

Saat mengonfigurasi bucket untuk menggunakan Kunci Bucket S3 untuk SSE-KMS pada objek baru, AWS KMS buat kunci tingkat ember yang digunakan untuk membuat kunci data unik untuk objek di bucket. Kunci Bucket S3 ini digunakan untuk jangka waktu terbatas dalam Amazon S3, mengurangi kebutuhan Amazon S3 untuk membuat permintaan AWS KMS untuk menyelesaikan operasi enkripsi.

Untuk informasi lebih lanjut tentang menggunakan Kunci Bucket S3, lihat Menggunakan Kunci Bucket Amazon S3.

Mengonfigurasi enkripsi default

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header

respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Bucket Amazon S3 mengaktifkan enkripsi bucket secara default, dan objek baru secara otomatis dienkripsi dengan menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3). Enkripsi ini berlaku untuk semua objek baru di bucket Amazon S3 Anda, dan Anda tidak akan dikenakan biaya.

Jika Anda memerlukan kontrol lebih besar atas kunci enkripsi Anda, seperti mengelola rotasi kunci dan hibah kebijakan akses, Anda dapat memilih untuk menggunakan enkripsi sisi server dengan AWS Key Management Service () kunci (SSE-KMS AWS KMS), atau enkripsi sisi server dua lapis dengan kunci (DSSE-KMS). AWS KMS Untuk informasi lebih lanjut tentang SSE-KMS, lihat Menentukan enkripsi di sisi server dengan AWS KMS (SSE-KMS). Untuk informasi lebih lanjut tentang DSSE-KMS, lihat the section called “Enkripsi di sisi server dua lapis (DSSE-KMS)”.

Jika Anda ingin menggunakan kunci KMS yang dimiliki oleh akun lain, Anda harus memiliki izin untuk menggunakan kunci tersebut. Untuk informasi selengkapnya tentang izin lintas akun untuk kunci KMS, lihat Membuat kunci KMS yang dapat digunakan oleh akun lain di Panduan Pengembang AWS Key Management Service .

Saat menyetel enkripsi bucket default ke SSE-KMS, Anda juga dapat mengonfigurasi Kunci Bucket S3 untuk mengurangi biaya permintaan. AWS KMS Untuk informasi selengkapnya, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3.

> **Note**
>
> Jika Anda menggunakan PutBucketEncryptionuntuk menyetel enkripsi bucket default ke SSE-KMS, Anda harus memverifikasi bahwa ID kunci KMS Anda sudah benar. Amazon S3 tidak memvalidasi ID kunci KMS yang disediakan dalam permintaan. PutBucketEncryption

Tidak ada biaya tambahan untuk menggunakan enkripsi default untuk bucket S3. Permintaan untuk mengonfigurasi perilaku enkripsi default dikenakan biaya permintaan standar Amazon S3. Untuk informasi tentang harga, lihat Harga Amazon S3. Untuk SSE-KMS dan DSSE-KMS, AWS KMS biaya berlaku dan tercantum pada harga.AWS KMS

Enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C) tidak didukung untuk enkripsi default.

Anda dapat mengonfigurasi enkripsi default Amazon S3 untuk bucket S3 dengan menggunakan konsol Amazon S3, SDK, Amazon S3 REST API AWS , dan (). AWS Command Line Interface AWS CLI

Perubahan yang perlu diingat sebelum mengaktifkan enkripsi default

Setelah Anda mengaktifkan enkripsi default untuk bucket, perilaku enkripsi berikut ini akan berlaku:

- Tidak ada perubahan pada enkripsi objek yang ada dalam bucket sebelum enkripsi default-nya diaktifkan.
- Saat Anda mengunggah objek setelah mengaktifkan enkripsi default:
  - Jika header permintaan PUT tidak mencakup informasi enkripsi, Amazon S3 akan menggunakan pengaturan enkripsi default bucket untuk mengenkripsi objek.
  - Jika header permintaan PUT mencakup informasi enkripsi, Amazon S3 menggunakan informasi enkripsi dari permintaan PUT untuk mengenkripsi objek sebelum menyimpannya di Amazon S3.
- Jika Anda menggunakan opsi SSE-KMS untuk konfigurasi enkripsi default, Anda tunduk pada kuota permintaan per detik (RPS) AWS KMS. Untuk informasi selengkapnya tentang kuota AWS KMS dan cara meminta kenaikan kuota, lihat Kuota di Panduan Pengembang AWS Key Management Service .

> **Note**
>
> Objek yang diunggah sebelum enkripsi default diaktifkan tidak akan dienkripsi. Untuk informasi tentang mengenkripsi objek yang ada, lihat the section called “Mengatur enkripsi bucket default”.

Menggunakan konsol S3

Untuk mengonfigurasi enkripsi default di bucket Amazon S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dari daftar Bucket, pilih nama bucket yang Anda inginkan.
4. Pilih tab Properti.

5. Di bagian bawah Enkripsi default, pilih Edit.
6. Untuk mengonfigurasi enkripsi default, di bawah Jenis enkripsi, pilih salah satu dari berikut ini:
   - Enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3)
   - Enkripsi sisi server dengan AWS Key Management Service kunci (SSE-KMS)
   - Enkripsi sisi server dua lapis dengan kunci (DSSE-KMS) AWS Key Management Service

   > **Important**
   >
   > Jika Anda menggunakan opsi SSE-KMS atau DSSE-KMS untuk konfigurasi enkripsi default, Anda tunduk pada kuota permintaan per detik (RPS) AWS KMS. Untuk informasi selengkapnya tentang AWS KMS kuota dan cara meminta kenaikan kuota, lihat Kuota di Panduan Pengembang AWS Key Management Service .

   Bucket dan objek baru dienkripsi secara default dengan SSE-S3, kecuali Anda menentukan jenis enkripsi default lain untuk bucket Anda. Untuk informasi selengkapnya tentang enkripsi default, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3.

   Untuk informasi selengkapnya tentang penggunaan enkripsi di sisi server Amazon S3 guna mengenkripsi data Anda, lihat Menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3).
7. Jika Anda memilih enkripsi sisi server dengan AWS Key Management Service kunci (SSE-KMS) atau enkripsi sisi server Dual-layer dengan kunci (DSSE-KMS), lakukan hal berikut: AWS Key Management Service

   a. Di bawah AWS KMS kunci, tentukan kunci KMS Anda dengan salah satu cara berikut ini:

      - Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari Anda AWS KMS keys, dan pilih kunci KMS Anda dari daftar kunci yang tersedia.

        Kunci Kunci yang dikelola AWS (`aws/s3`) dan kunci terkelola pelanggan Anda muncul dalam daftar ini. Untuk informasi selengkapnya tentang kunci yang dikelola pelanggan, lihat Kunci dan AWS kunci pelanggan di Panduan AWS Key Management Service Pengembang.
      - Untuk memasukkan ARN kunci KMS, pilih Masukkan AWS KMS key ARN, dan masukkan ARN kunci KMS Anda di bidang yang muncul.

- Untuk membuat kunci terkelola pelanggan baru di AWS KMS konsol, pilih Buat kunci KMS.

  Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat kunci di Panduan AWS Key Management Service Pengembang.

> **Important**
>
> Anda hanya dapat menggunakan tombol KMS yang diaktifkan Wilayah AWS sama dengan bucket. Saat memilih Pilih dari kunci KMS Anda, konsol S3 hanya mencantumkan 100 kunci KMS per Wilayah. Jika Anda memiliki lebih dari 100 tombol KMS di Wilayah yang sama, Anda hanya dapat melihat 100 kunci KMS pertama di konsol S3. Untuk menggunakan kunci KMS yang tidak terdaftar di konsol, pilih Masukkan ARN AWS KMS key , dan masukkan ARN kunci KMS. Saat Anda menggunakan enkripsi sisi server AWS KMS key untuk Amazon S3, Anda harus memilih kunci KMS enkripsi simetris. Amazon S3 hanya mendukung kunci KMS enkripsi simetris. Untuk informasi selengkapnya terkait kunci ini, lihat Membuat kunci enkripsi simetris KMS dalam Panduan Pengembang AWS Key Management Service .

Untuk informasi selengkapnya tentang penggunaan SSE-KMS dengan Amazon S3, lihat Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS). Untuk informasi selengkapnya tentang DSSE-KMS, lihat the section called “Enkripsi di sisi server dua lapis (DSSE-KMS)”.

b. Saat Anda mengonfigurasikan bucket untuk menggunakan enkripsi default dengan SSE-KMS, Anda juga dapat mengaktifkan Kunci Bucket S3. S3 Bucket Keys menurunkan biaya enkripsi dengan mengurangi lalu lintas permintaan dari Amazon S3 ke. AWS KMS Untuk informasi selengkapnya, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3.

   Untuk menggunakan Kunci Bucket S3, di bagian bawah Kunci Bucket, pilih Aktifkan.

> **Note**
>
> Kunci Bucket S3 tidak didukung untuk DSSE-KMS.

8. Pilih Simpan perubahan.

Menggunakan AWS CLI

Contoh ini menunjukkan cara mengonfigurasi enkripsi default dengan menggunakan SSE-S3 atau dengan menggunakan SSE-KMS dengan Kunci Bucket S3.

Untuk informasi selengkapnya tentang enkripsi default, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3. Untuk informasi selengkapnya tentang menggunakan enkripsi default AWS CLI untuk mengonfigurasi, lihat put-bucket-encryption.

Example – Enkripsi default dengan SSE-S3

Contoh ini mengonfigurasi enkripsi bucket default dengan kunci terkelola Amazon S3.

```
aws s3api put-bucket-encryption --bucket example-s3-bucket --server-side-encryption-
configuration '{
    "Rules": [
        {
            "ApplyServerSideEncryptionByDefault": {
                "SSEAlgorithm": "AES256"
            }
        }
    ]
}'
```

Example – Enkripsi default dengan SSE-KMS menggunakan Kunci Bucket S3

Contoh ini mengonfigurasi enkripsi bucket default dengan SSE-KMS menggunakan Kunci Bucket S3.

```
aws s3api put-bucket-encryption --bucket example-s3-bucket --server-side-encryption-
configuration '{
    "Rules": [
            {
                "ApplyServerSideEncryptionByDefault": {
                    "SSEAlgorithm": "aws:kms",
                    "KMSMasterKeyID": "KMS-Key-ARN"
                },
                "BucketKeyEnabled": true
            }
        ]
    }'
```

Penggunaan API REST

Gunakan operasi `PutBucketEncryption` API REST untuk mengaktifkan enkripsi default, dan untuk menyetel jenis enkripsi di sisi server yang akan menggunakan—SSE-S3, SSE-KMS, atau DSSE-KMS.

Untuk informasi selengkapnya, lihat PutBucketEncryption dalam Referensi API Amazon Simple Storage Service.

Memantau enkripsi default dengan AWS CloudTrail dan Amazon EventBridge

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Anda dapat melacak permintaan konfigurasi enkripsi default untuk bucket Amazon S3 menggunakan peristiwa AWS CloudTrail . Nama peristiwa API berikut digunakan dalam CloudTrail log:

- `PutBucketEncryption`
- `GetBucketEncryption`
- `DeleteBucketEncryption`

Anda juga dapat membuat EventBridge aturan untuk mencocokkan CloudTrail peristiwa untuk panggilan API ini. Untuk informasi selengkapnya tentang CloudTrail acara, lihatMengaktifkan pencatatan untuk objek dalam bucket menggunakan konsol tersebut. Untuk informasi selengkapnya tentang EventBridge acara, lihat Acara dari Layanan AWS.

Anda dapat menggunakan CloudTrail log untuk tindakan Amazon S3 tingkat objek untuk `PUT` melacak `POST` dan meminta ke Amazon S3. Anda dapat menggunakan tindakan ini untuk

memverifikasi apakah enkripsi default digunakan untuk mengenkripsi objek saat permintaan PUT yang masuk tidak memiliki header enkripsi.

Saat Amazon S3 mengenkripsi objek menggunakan pengaturan enkripsi default, log mencakup salah satu bidang berikut sebagai pasangan nama-nilai: `"SSEApplied":"Default_SSE_S3"`, `"SSEApplied":"Default_SSE_KMS"`, atau `"SSEApplied":"Default_DSSE_KMS"`.

Saat Amazon S3 mengenkripsi objek menggunakan header enkripsi PUT, log mencakup salah satu bidang berikut sebagai pasangan nama-nilai: `"SSEApplied":"SSE_S3"`, `"SSEApplied":"SSE_KMS"`, `"SSEApplied":"DSSE_KMS"`, atau `"SSEApplied":"SSE_C"`.

Untuk pengunggahan multibagian, informasi ini disertakan dalam permintaan operasi `InitiateMultipartUpload` API Anda. Untuk informasi lebih lanjut tentang menggunakan CloudTrail dan CloudWatch, lihatPemantauan Amazon S3.

Bekerja dengan Mountpoint untuk Amazon S3

Mountpoint untuk Amazon S3 adalah klien file open source throughput tinggi untuk memasang bucket Amazon S3 sebagai sistem file lokal. Dengan Mountpoint, aplikasi Anda dapat mengakses objek yang disimpan di Amazon S3 melalui operasi sistem file, seperti membuka dan membaca. Mountpoint secara otomatis menerjemahkan operasi ini ke dalam panggilan API objek S3, memberikan aplikasi Anda akses ke penyimpanan elastis serta throughput Amazon S3 melalui antarmuka file.

Mountpoint untuk Amazon S3 umumnya tersedia untuk penggunaan produksi pada aplikasi pembacaan berat berskala besar Anda: danau data, pelatihan machine learning, rendering gambar, simulasi kendaraan otonom, extract, transform, and load (ETL), dan banyak lagi.

Mountpoint mendukung operasi sistem file dasar, dan dapat membaca file berukuran hingga 5 TB. Mountpoint dapat mendaftarkan dan membaca file yang ada, dan dapat membuat file yang baru. Mountpoint tidak dapat memodifikasi file yang ada atau menghapus direktori, dan tidak mendukung tautan simbolis atau penguncian file. Mountpoint ideal untuk aplikasi yang tidak memerlukan semua fitur sistem file bersama dan izin bergaya POSIX, namun memerlukan throughput elastis Amazon S3 untuk membaca dan menulis set data S3 yang besar. Untuk detailnya, lihat Perilaku sistem file Mountpoint pada GitHub. Untuk beban kerja yang memerlukan dukungan penuh POSIX, kami merekomendasikan Amazon FSx for Lustre dan dukungan untuk menautkan bucket S3.

Mountpoint untuk Amazon S3 hanya tersedia untuk sistem operasi Linux. Anda dapat menggunakan Mountpoint untuk mengakses Objek S3 di semua kelas penyimpanan kecuali S3 Glacier Flexible

Retrieval, S3 Glacier Deep Archive, S3 Intelligent-Tiering Archive Access Tier, dan S3 Intelligent-Tiering Deep Archive Access Tier.

Topik

- Instalasi Mountpoint
- Mengkonfigurasi dan menggunakan Mountpoint

Instalasi Mountpoint

Anda dapat mengunduh dan menginstal paket Mountpoint untuk Amazon S3 dengan menggunakan baris perintah. Petunjuk untuk mengunduh dan menginstal Mountpoint bervariasi, tergantung pada sistem operasi Linux yang Anda gunakan.

Topik

- Distribusi berbasis RPM (Amazon Linux, Fedora, CentOS, RHEL)
- Distribusi berbasis DEB (Debian, Ubuntu)
- Distribusi Linux lainnya
- Memverifikasi tanda tangan paket Mountpoint untuk Amazon S3

Distribusi berbasis RPM (Amazon Linux, Fedora, CentOS, RHEL)

1. Salin URL unduhan untuk arsitektur Anda.

   x86_64:

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.rpm
   ```

   ARM64 (Graviton):

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/arm64/mount-s3.rpm
   ```

2. Unduh paket Mountpoint untuk Amazon S3. Ganti *download-link* dengan URL unduhan yang sesuai dari langkah sebelumnya.

   ```
   wget download-link
   ```

3. (Opsional) Verifikasi keaslian dan integritas file yang diunduh. Pertama, salin URL tanda tangan untuk arsitektur Anda.

   x86_64:

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.rpm.asc
   ```

   ARM64 (Graviton):

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/arm64/mount-s3.rpm.asc
   ```

   Selanjutnya, lihat Memverifikasi tanda tangan paket Mountpoint untuk Amazon S3.

4. Instal paket menggunakan perintah berikut ini:

   ```
   sudo yum install ./mount-s3.rpm
   ```

5. Verifikasi bahwa Mountpoint berhasil diinstal dengan memasukkan perintah berikut:

   ```
   mount-s3 --version
   ```

   Anda akan melihat output yang serupa dengan yang berikut:

   ```
   mount-s3 1.3.1
   ```

Distribusi berbasis DEB (Debian, Ubuntu)

1. Salin URL unduhan untuk arsitektur Anda.

   x86_64:

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.deb
   ```

   ARM64 (Graviton):

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/arm64/mount-s3.deb
   ```

2. Unduh paket Mountpoint untuk Amazon S3. Ganti *download-link* dengan URL unduhan yang sesuai dari langkah sebelumnya.

```
wget download-link
```

3. (Opsional) Verifikasi keaslian dan integritas file yang diunduh. Pertama, salin URL tanda tangan untuk arsitektur Anda.

   x86_64:

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.deb.asc
   ```

   ARM64 (Graviton):

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/arm64/mount-s3.deb.asc
   ```

   Selanjutnya, lihat Memverifikasi tanda tangan paket Mountpoint untuk Amazon S3.

4. Instal paket menggunakan perintah berikut ini:

   ```
   sudo apt-get install ./mount-s3.deb
   ```

5. Konfirmasikan apakah Mountpoint untuk Amazon S3 berhasil diinstal dengan menjalankan perintah berikut ini:

   ```
   mount-s3 --version
   ```

   Anda akan melihat output yang serupa dengan yang berikut:

   ```
   mount-s3 1.3.1
   ```

Distribusi Linux lainnya

1. Konsultasikan dokumentasi sistem operasi Anda untuk menginstal paket `FUSE` dan `libfuse2` yang diperlukan.
2. Salin URL unduhan untuk arsitektur Anda.

   x86_64:

   ```
   https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.tar.gz
   ```

ARM64 (Graviton):

```
https://s3.amazonaws.com/mountpoint-s3-release/latest/arm64/mount-s3.tar.gz
```

3. Unduh paket Mountpoint untuk Amazon S3. Ganti *download-link* dengan URL unduhan yang sesuai dari langkah sebelumnya.

```
wget download-link
```

4. (Opsional) Verifikasi keaslian dan integritas file yang diunduh. Pertama, salin URL tanda tangan untuk arsitektur Anda.

   x86_64:

```
https://s3.amazonaws.com/mountpoint-s3-release/latest/x86_64/mount-s3.tar.gz.asc
```

   ARM64 (Graviton):

```
https://s3.amazonaws.com/mountpoint-s3-release/latest/arm64/mount-s3.tar.gz.asc
```

   Selanjutnya, lihat Memverifikasi tanda tangan paket Mountpoint untuk Amazon S3.

5. Instal paket menggunakan perintah berikut ini:

```
sudo mkdir -p /opt/aws/mountpoint-s3 && sudo tar -C /opt/aws/mountpoint-s3 -xzf ./
mount-s3.tar.gz
```

6. Tambahkan biner `mount-s3` ke variabel lingkungan `PATH` Anda. Dalam file `$HOME/.profile` Anda, tambahkan baris berikut:

```
export PATH=$PATH:/opt/aws/mountpoint-s3/bin
```

   Simpan file `.profile`, dan jalankan perintah berikut ini:

```
source $HOME/.profile
```

7. Konfirmasikan apakah Mountpoint untuk Amazon S3 berhasil diinstal dengan menjalankan perintah berikut ini:

```
mount-s3 --version
```

Anda akan melihat output yang serupa dengan yang berikut:

```
mount-s3 1.3.1
```

Memverifikasi tanda tangan paket Mountpoint untuk Amazon S3

1. Instal GnuPG (perintah gpg). Ini diperlukan untuk memverifikasi keaslian dan integritas paket Mountpoint untuk Amazon S3 yang diunduh. GnuPG diinstal secara default di Amazon Linux Amazon Machine Images (AMI). Setelah Anda menginstal GnuPG, lanjutkan ke langkah 2.
2. Unduh kunci publik Mountpoint dengan menjalankan perintah berikut ini:

   ```
   wget https://s3.amazonaws.com/mountpoint-s3-release/public_keys/KEYS
   ```

3. Impor kunci publik Mountpoint ke dalam keyring Anda dengan menjalankan perintah berikut:

   ```
   gpg --import KEYS
   ```

4. Verifikasi sidik jari kunci publik Mountpoint dengan menjalankan perintah berikut:

   ```
   gpg --fingerprint mountpoint-s3@amazon.com
   ```

   Konfirmasikan bahwa string sidik jari yang ditampilkan cocok dengan yang berikut:

   ```
   673F E406 1506 BB46 9A0E  F857 BE39 7A52 B086 DA5A
   ```

   Jika string sidik jarinya tidak cocok, hentikan penginstalan Mountpoint, dan hubungi AWS Support.

5. Unduh file tanda tangan paket. Ganti *signature-link* dengan tautan tanda tangan yang sesuai dari bagian sebelumnya.

   ```
   wget signature-link
   ```

6. Verifikasi tanda tangan paket yang diunduh dengan menjalankan perintah berikut. Ganti *signature-filename* dengan nama file dari langkah sebelumnya.

```
gpg --verify signature-filename
```

Misalnya, pada distribusi berbasis RPM, termasuk Amazon Linux, masukkan perintah berikut:

```
gpg --verify mount-s3.rpm.asc
```

7. Outputnya harus menyertakan frasa `Good signature`. Jika output menyertakan frasa `BAD signature`, unduh ulang file paket Mountpoint dan ulangi langkah-langkah ini. Jika masalahnya berlanjut, hentikan penginstalan Mountpoint, dan hubungi AWS Support.

   Outputnya mungkin termasuk peringatan tentang tanda tangan tepercaya. Ini tidak menunjukkan masalah. Ini hanya berarti bahwa Anda belum memverifikasi kunci publik Mountpoint secara independen.

Mengkonfigurasi dan menggunakan Mountpoint

Untuk menggunakan Mountpoint untuk Amazon S3, host Anda memerlukan kredensyal yang AWS valid dengan akses ke bucket atau bucket yang ingin Anda pasang. Untuk berbagai macam cara mengautentikasi, lihat Kredensial AWS Mountpoint pada GitHub.

Misalnya, Anda dapat membuat pengguna dan peran baru AWS Identity and Access Management (IAM) untuk tujuan ini. Pastikan bahwa peran ini memiliki akses ke bucket, atau bucket-bucket yang ingin Anda pasang. Anda dapat meneruskan peran IAM ke instans Amazon EC2 Anda dengan profil instans.

Menggunakan Mountpoint untuk Amazon S3

Gunakan Mountpoint untuk Amazon S3 untuk melakukan hal berikut ini:

1. Pasang bucket dengan perintah `mount-s3`.

   Dalam contoh berikut, ganti *DOC-EXAMPLE-BUCKET* dengan nama bucket S3 Anda, dan ganti *~/mnt* dengan direktori di host tempat Anda ingin bucket S3 Anda dipasang.

   ```
   mkdir ~/mnt
   mount-s3 DOC-EXAMPLE-BUCKET ~/mnt
   ```

Karena klien Mountpoint berjalan di latar belakang secara default, direktori *~/mnt* akan memberikan Anda akses ke objek di bucket S3 Anda.

2. Mengakses objek di dalam bucket Anda melalui Mountpoint.

   Setelah memasang bucket secara lokal, Anda dapat menggunakan perintah umum Linux, seperti `cat` atau `ls`, untuk bekerja dengan Objek S3. Mountpoint untuk Amazon S3 menafsirkan kunci di bucket S3 Anda sebagai jalur sistem file dengan memisahkannya pada karakter garis miring ke depan (`/`). Misalnya, jika Anda memiliki kunci objek `Data/2023-01-01.csv` di dalam bucket Anda, Anda akan memiliki direktori bernama `Data` dalam sistem file Mountpoint Anda, dengan file bernama `2023-01-01.csv` di dalamnya.

   Mountpoint untuk Amazon S3 secara sengaja tidak menerapkan spesifikasi standar POSIX lengkap untuk sistem file. Mountpoint dioptimalkan untuk beban kerja yang memerlukan akses baca dan tulis throughput tinggi ke data yang disimpan di Amazon S3 melalui antarmuka sistem file, namun tidak bergantung pada fitur sistem file. Untuk informasi selengkapnya, lihat perilaku sistem file Mountpoint untuk Amazon S3 pada GitHub. Pelanggan yang membutuhkan semantik sistem file yang lebih kaya harus mempertimbangkan layanan AWS file lainnya, seperti Amazon Elastic File System (Amazon EFS) atau Amazon FSx.

3. Lepaskan bucket Anda dengan menggunakan perintah `umount`. Perintah ini melepaskan bucket S3 Anda, dan keluar dari Mountpoint.

   Untuk menggunakan perintah contoh berikut ini, ganti *~/mnt* dengan direktori di host tempat bucket S3 Anda dipasang.

   ```
   umount ~/mnt
   ```

   > **Note**
   >
   > Untuk mendapatkan daftar opsi untuk perintah ini, jalankan `umount --help`.

Untuk detail konfigurasi Mountpoint tambahan, lihat Konfigurasi bucket S3, dan konfigurasi sistem file pada GitHub.

Mengonfigurasi caching di Mountpoint

Saat Anda menggunakan Mountpoint untuk Amazon S3, Anda dapat mengonfigurasinya untuk menyimpan cache data yang terakhir diakses dari bucket S3 Anda di penyimpanan instans Amazon EC2, atau volume Amazon EBS yang terlampir. Caching data ini dapat membantu mempercepat performa, dan mengurangi biaya akses data berulang. Caching di Mountpoint sangat ideal untuk kasus penggunaan di mana Anda berulang kali membaca data yang sama yang tidak berubah selama beberapa kali pembacaan. Misalnya, Anda dapat menggunakan caching dengan tugas pelatihan machine learning yang perlu membaca set data pelatihan beberapa kali untuk meningkatkan akurasi model.

Saat memasang bucket S3, Anda dapat mengaktifkan caching melalui flag secara opsional. Anda dapat mengonfigurasi lokasi dan ukuran cache data serta lamanya metadata disimpan dalam cache. Saat Anda memasang bucket dan caching diaktifkan, Mountpoint membuat subdirektori kosong di lokasi cache yang dikonfigurasi, jika subdirektori tersebut belum ada. Saat Anda pertama kali memasang bucket dan melepasnya, Mountpoint menghapus konten lokasi cache. Untuk informasi selengkapnya tentang mengonfigurasi dan menggunakan caching di Mountpoint, lihat konfigurasi Caching Mountpoint for Amazon S3 aktif. GitHub

Saat Anda memasang bucket S3, Anda dapat mengaktifkan caching dengan flag `--cache` *`CACHE_PATH`*. Dalam contoh berikut, ganti *`CACHE_PATH`* dengan filepath ke direktori tempat Anda ingin menyimpan data Anda. Ganti *`DOC-EXAMPLE-BUCKET`* dengan nama bucket S3 Anda, dan ganti *`~/mnt`* dengan direktori di host tempat Anda ingin bucket S3 Anda dipasang.

```
mkdir ~/mnt
mount-s3 --cache CACHE_PATH DOC-EXAMPLE-BUCKET ~/mnt
```

> **Important**
>
> Jika Anda mengaktifkan caching, Mountpoint akan mempertahankan konten objek yang tidak terenkripsi dari bucket S3 Anda di lokasi caching yang dikonfigurasi saat pemasangan. Untuk melindungi data Anda, sebaiknya Anda membatasi akses ke lokasi cache data.

Pemecahan Masalah Mountpoint

Mountpoint untuk Amazon S3 didukung oleh. AWS Support Jika Anda membutuhkan bantuan, hubungi AWS Support Pusat.

Anda juga dapat meninjau dan mengirimkan Masalah Mountpoint di GitHub.

Jika Anda menemukan potensi masalah keamanan dalam proyek ini, kami meminta Anda untuk memberi tahu Keamanan AWS melalui halaman pelaporan kerentanan kami. Jangan membuat pelaporannya di masalah GitHub publik.

Jika aplikasi Anda berperilaku tidak terduga dengan Mountpoint, Anda dapat memeriksa informasi log untuk mendiagnosis masalahnya.

Pencatatan log

Secara default, Mountpoint mengeluarkan informasi log tingkat keparahan tinggi ke syslog.

Untuk melihat log pada sebagian besar distribusi Linux modern, termasuk Amazon Linux, jalankan perintah `journald` berikut:

```
journalctl -e SYSLOG_IDENTIFIER=mount-s3
```

Pada sistem Linux lain, entri `syslog` kemungkinan akan ditulis ke file seperti `/var/log/syslog`.

Anda dapat menggunakan log ini untuk memecahkan masalah aplikasi Anda. Misalnya, jika aplikasi Anda mencoba menimpa file yang sudah ada, operasi tersebut gagal, dan Anda akan melihat baris yang mirip dengan berikut ini di dalam log:

```
[WARN] open{req=12 ino=2}: mountpoint_s3::fuse: open failed: inode error: inode 2 (full
 key "README.md") is not writable
```

Untuk informasi selengkapnya, lihat Pencatatan log Mountpoint untuk Amazon S3 pada GitHub.

Mengonfigurasi transfer file yang cepat dan aman menggunakan Amazon S3 Transfer Acceleration

Akselerasi Transfer Amazon S3 adalah fitur tingkat bucket yang memungkinkan pentransferan file dengan cepat, mudah, dan aman dalam jarak jauh antara klien Anda dan bucket S3. Transfer Acceleration dirancang untuk mengoptimalkan kecepatan transfer dari seluruh dunia ke dalam bucket S3. Transfer Acceleration memanfaatkan lokasi edge yang didistribusikan secara global di Amazon CloudFront. Saat datanya tiba di lokasi edge, data tersebut diarahkan ke Amazon S3 melalui jalur jaringan yang dioptimalkan.

Saat Anda menggunakan Transfer Acceleration, biaya transfer data tambahan mungkin berlaku. Untuk informasi selengkapnya tentang harga, lihat Harga Amazon S3.

Mengapa menggunakan Percepatan Transfer?

Anda mungkin ingin menggunakan Transfer Acceleration pada bucket karena berbagai alasan:

- Pelanggan Anda mengunggah ke bucket terpusat dari seluruh dunia.
- Anda mentransfer ukuran data gigabyte ke terabyte secara teratur di seluruh benua.
- Anda tidak dapat menggunakan semua bandwidth yang tersedia melalui internet saat mengunggah ke Amazon S3.

Untuk informasi lebih lanjut tentang kapan menggunakan Transfer Acceleration, lihat FAQ Amazon S3.

Persyaratan untuk menggunakan Transfer Acceleration

Hal berikut ini diperlukan saat Anda menggunakan Transfer Acceleration pada bucket S3:

- Transfer Acceleration hanya didukung pada permintaan bergaya hosting virtual. Untuk informasi selengkapnya tentang permintaan bergaya hosting virtual, lihat Membuat permintaan menggunakan API REST.
- Nama bucket yang digunakan untuk Transfer Acceleration harus mematuhi DNS, dan tidak boleh mengandung titik (".").
- Transfer Acceleration harus diaktifkan pada bucket. Untuk informasi selengkapnya, lihat Mengaktifkan dan menggunakan S3 Transfer Acceleration.

  Setelah Anda mengaktifkan Transfer Acceleration pada bucket, mungkin diperlukan waktu hingga 20 menit sebelum kecepatan transfer data ke bucket tersebut meningkat.

> **Note**
>
> Transfer Acceleration saat ini tidak didukung untuk bucket yang terletak di Wilayah berikut:
>
> - Asia Pasifik (Tokyo) (ap-northeast-1)
> - Asia Pasifik (Seoul) (ap-northeast-2)
> - Asia Pasifik (Mumbai) (ap-south-1)
> - Asia Pasifik (Singapura) (ap-southeast-1)

- Asia Pasifik (Sydney) (ap-southeast-2)
- Kanada (Pusat) (ca-central-1)
- Eropa (Frankfurt) (eu-central-1)
- Eropa (Irlandia) (eu-west-1)
- Eropa (London) (eu-west-2)
- Eropa (Paris) (eu-west-3)
- Amerika Selatan (Sao Paulo) (sa-east-1)
- AS Timur (Virginia Utara) (us-east-1)
- AS Timur (Ohio) (us-east-2)
- AS Barat (California Utara) (us-west-1)
- AS Barat (Oregon) (us-west-2)

- Untuk mengakses bucket yang diaktifkan untuk Transfer Acceleration, Anda harus menggunakan titik akhir `bucketname.s3-accelerate.amazonaws.com`. Atau, gunakan titik akhir dual-stack `bucketname.s3-accelerate.dualstack.amazonaws.com` untuk menyambung ke bucket yang diaktifkan melalui IPv6. Anda dapat terus menggunakan titik akhir reguler untuk transfer data standar.
- Anda harus menjadi pemilik bucket untuk mengatur kondisi akselerasi transfer. Pemilik bucket dapat menetapkan izin kepada pengguna lain untuk memungkinkan mereka mengatur kondisi akselerasi pada bucket. `s3:PutAccelerateConfiguration` mengizinkan pengguna untuk mengaktifkan atau menonaktifkan Transfer Acceleration pada bucket. `s3:GetAccelerateConfiguration`Izin ini memungkinkan pengguna untuk mengembalikan status Akselerasi Transfer bucket, yaitu salah satu atau `Enabled` `Suspended`.

Bagian berikut ini menjelaskan bagaimana caranya untuk memulai dan menggunakan Amazon S3 Transfer Acceleration untuk mentransfer data.

Topik

- Memulai Amazon S3 Transfer Acceleration
- Mengaktifkan dan menggunakan S3 Transfer Acceleration
- Menggunakan Alat Perbandingan Kecepatan Amazon S3 Transfer Acceleration

Memulai Amazon S3 Transfer Acceleration

Amazon S3 Transfer Acceleration memungkinkan transfer file yang cepat, mudah, dan aman melalui jarak jauh antara klien Anda dan bucket S3. Transfer Acceleration menggunakan lokasi edge yang didistribusikan secara global di Amazon CloudFront. Saat datanya tiba di lokasi edge, data diarahkan ke Amazon S3 melalui jalur jaringan yang dioptimalkan.

Untuk mulai menggunakan Amazon S3 Transfer Acceleration, lakukan langkah-langkah berikut:

1. Aktifkan Transfer Acceleration pada bucket

   Anda dapat mengaktifkan Transfer Acceleration pada bucket dengan cara berikut ini:

   - Gunakan konsol Amazon S3.
   - Gunakan operasi API REST PUT Bucket dipercepat.
   - Gunakan AWS CLI dan AWS SDK. Untuk informasi selengkapnya, lihat Mengembangkan dengan Amazon S3 menggunakan SDK AWS.

   Untuk informasi selengkapnya, lihat Mengaktifkan dan menggunakan S3 Transfer Acceleration.

   > **Note**
   >
   > Agar bucket Anda berfungsi dengan akselerasi transfer, nama bucket harus sesuai dengan persyaratan penamaan DNS, dan tidak boleh berisi titik (".").

2. Transfer data ke dan dari bucket yang diaktifkan dengan akselerasi

   Gunakan salah satu dari nama domain titik akhir s3-accelerate berikut ini:

   - Untuk mengakses bucket yang mendukung akselerasi, gunakan `bucketname.s3-accelerate.amazonaws.com`.
   - Untuk mengakses bucket yang mendukung akselerasi melalui IPv6, gunakan `bucketname.s3-accelerate.dualstack.amazonaws.com`.

     Titik akhir tumpukan ganda Amazon S3 mendukung permintaan ke bucket S3 melalui IPv6 dan IPv4. Titik akhir dual-stack Transfer Acceleration hanya menggunakan jenis nama titik akhir yang dihosting bergaya virtual. Untuk informasi lebih lanjut, lihat Memulai membuat permintaan melalui IPv6 dan Menggunakan titik akhir tumpukan ganda Amazon S3.

> **Note**
>
> Aplikasi transfer data Anda harus menggunakan salah satu dari dua jenis titik akhir berikut untuk mengakses bucket agar transfer data lebih cepat: `.s3-accelerate.amazonaws.com` atau `.s3-accelerate.dualstack.amazonaws.com` untuk titik akhir tumpukan ganda. Anda dapat terus menggunakan titik akhir reguler untuk transfer data standar.

Anda dapat mengarahkan objek Amazon S3 PUT dan permintaan objek GET ke nama domain titik akhir `s3-accelerate` setelah Anda mengaktifkan Transfer Acceleration. Misalnya, bayangkan bahwa saat ini Anda memiliki aplikasi API REST menggunakan PUT Objek yang menggunakan nama host `mybucket.s3.us-east-1.amazonaws.com` di permintaan PUT. Untuk mempercepat PUT, Anda mengubah nama host dalam permintaan Anda ke `mybucket.s3-accelerate.amazonaws.com`. Untuk kembali menggunakan kecepatan pengunggahan standar, ubah namanya kembali ke `mybucket.s3.us-east-1.amazonaws.com`.

Setelah Transfer Acceleration diaktifkan, mungkin memerlukan waktu hingga 20 menit untuk mewujudkan manfaat performanya. Namun, titik akhir akselerasi tersedia segera setelah Anda mengaktifkan Transfer Acceleration.

Anda dapat menggunakan titik akhir percepatan di AWS CLI, AWS SDK, dan alat lain yang mentransfer data ke dan dari Amazon S3. Jika Anda menggunakan AWS SDK, beberapa bahasa yang didukung menggunakan flag konfigurasi klien titik akhir percepatan sehingga Anda tidak perlu secara eksplisit menyetel titik akhir untuk Transfer Acceleration ke. *`bucketname`*`.s3-accelerate.amazonaws.com` Untuk contoh cara menggunakan flag konfigurasi klien titik akhir yang dipercepat, lihat Mengaktifkan dan menggunakan S3 Transfer Acceleration.

Anda dapat menggunakan semua operasi Amazon S3 melalui titik akhir percepatan transfer kecuali untuk hal berikut ini:

- Layanan GET (mencantumkan bucket)
- Bucket PUT (membuat bucket)
- Bucket DELETE

Lalu, Amazon S3 Transfer Acceleration juga tidak mendukung salinan lintas wilayah menggunakan PUT Objek-Salin.

Mengaktifkan dan menggunakan S3 Transfer Acceleration

Amazon S3 Transfer Acceleration memungkinkan transfer file yang cepat, mudah, dan aman melalui jarak jauh antara klien Anda dan bucket S3. Anda dapat mengaktifkan Transfer Acceleration menggunakan konsol S3, AWS Command Line Interface (AWS CLI), API, atau AWS SDK.

Bagian ini memberikan contoh cara mengaktifkan Amazon S3 Transfer Acceleration pada bucket dan menggunakan titik akhir akselerasi untuk bucket yang diaktifkan.

Untuk informasi lebih lanjut tentang persyaratan Transfer Acceleration, lihat Mengonfigurasi transfer file yang cepat dan aman menggunakan Amazon S3 Transfer Acceleration.

Menggunakan konsol S3

> **Note**
>
> Jika Anda ingin membandingkan kecepatan pengunggahan yang dipercepat dan yang tidak dipercepat, buka Alat Perbandingan Kecepatan Amazon S3 Transfer Acceleration.
> Alat Perbandingan Kecepatan menggunakan unggahan multibagian untuk mentransfer file dari browser Anda ke berbagai Wilayah AWS dengan dan tanpa akselerasi transfer Amazon S3. Anda dapat membandingkan kecepatan unggahan untuk pengunggahan langsung dan mentransfer unggahan yang dipercepat oleh Wilayah.

Cara mengaktifkan akselerasi transfer untuk bucket S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Bucket, pilih nama bucket yang ingin Anda aktifkan untuk akselerasi transfer.
3. Pilih Properti.
4. Di bagian bawah Akselerasi transfer, pilih Edit.
5. Pilih Aktifkan, dan pilih Simpan perubahan.

Untuk mengakses transfer data yang dipercepat

1. Setelah Amazon S3 mengaktifkan akselerasi transfer untuk bucket Anda, lihat tab Properti untuk bucket tersebut.
2. Di bagian bawah Akselerasi transfer, Titik akhir yang dipercepat menampilkan titik akhir akselerasi transfer untuk bucket Anda. Gunakan titik akhir ini untuk mengakses transfer data yang dipercepat ke dan dari bucket Anda.

   Jika Anda menangguhkan akselerasi transfer, titik akhir percepatan tidak lagi bekerja.

Menggunakan AWS CLI

Berikut ini adalah contoh AWS CLI perintah yang digunakan untuk Transfer Acceleration. Untuk petunjuk tentang pengaturan AWS CLI, lihatMengembangkan dengan Amazon S3 menggunakan AWS CLI.

Mengaktifkan Transfer Acceleration pada bucket

Gunakan AWS CLI put-bucket-accelerate-configurationperintah untuk mengaktifkan atau menangguhkan Transfer Acceleration pada bucket.

Contoh berikut menetapkan `Status=Enabled` untuk mengaktifkan Transfer Acceleration pada bucket. Anda menggunakan `Status=Suspended` untuk menangguhkan Transfer Acceleration.

Example

```
$ aws s3api put-bucket-accelerate-configuration --bucket bucketname --accelerate-
configuration Status=Enabled
```

Menggunakan Transfer Acceleration

Anda dapat mengarahkan semua permintaan Amazon S3 yang dibuat oleh AWS CLI perintah s3 dan s3api ke titik akhir percepatan:. `s3-accelerate.amazonaws.com` Untuk melakukan ini, atur nilai konfigurasi `use_accelerate_endpoint` ke `true` dalam profil di file AWS Config Anda. Transfer Acceleration harus diaktifkan pada bucket Anda untuk menggunakan titik akhir akselerasi.

Semua permintaan dikirim menggunakan pengalamatan bucket bergaya virtual: `my-bucket.s3-accelerate.amazonaws.com`. Setiap permintaan `ListBuckets`, `CreateBucket`, dan `DeleteBucket` tidak dikirim ke titik akhir percepatan karena titik akhir tidak mendukung operasi tersebut.

Untuk informasi selengkapnya tentang `use_accelerate_endpoint`, lihat AWS CLI Konfigurasi S3 dalam AWS CLI Referensi Perintah.

Contoh berikut ini menetapkan `use_accelerate_endpoint` ke `true` di profil default.

Example

```
$ aws configure set default.s3.use_accelerate_endpoint true
```

Jika Anda ingin menggunakan titik akhir percepatan untuk beberapa AWS CLI perintah tetapi tidak yang lain, Anda dapat menggunakan salah satu dari dua metode berikut:

- Gunakan titik akhir akselerasi untuk perintah s3 atau s3api dengan menyetel parameter `--endpoint-url` ke `https://s3-accelerate.amazonaws.com`.
- Siapkan profil terpisah di file AWS Config Anda. Misalnya, buat satu profil yang mengatur `use_accelerate_endpoint` hingga `true` dan profil yang tidak mengatur `use_accelerate_endpoint`. Ketika Anda menjalankan perintah, tentukan profil mana yang ingin Anda gunakan, tergantung pada apakah Anda ingin menggunakan titik akhir tumpukan ganda atau tidak.

Mengunggah objek ke bucket yang diaktifkan untuk Transfer Acceleration

Contoh berikut ini mengunggah file ke bucket yang diaktifkan untuk Transfer Acceleration, dengan menggunakan profil default yang telah dikonfigurasi untuk menggunakan titik akhir percepatan.

Example

```
$ aws s3 cp file.txt s3://bucketname/keyname --region region
```

Contoh berikut ini mengunggah file ke bucket yang diaktifkan untuk Transfer Acceleration dengan menggunakan `--endpoint-url` untuk menentukan titik akhir akselerasi.

Example

```
$ aws configure set s3.addressing_style virtual
$ aws s3 cp file.txt s3://bucketname/keyname --region region --endpoint-url https://s3-
accelerate.amazonaws.com
```

Menggunakan AWS SDK

Berikut ini adalah contoh penggunaan Transfer Acceleration untuk mengunggah objek ke Amazon S3 menggunakan SDK. AWS *Beberapa bahasa yang didukung AWS SDK (misalnya, Java dan .NET) menggunakan flag konfigurasi klien endpoint percepatan sehingga Anda tidak perlu secara eksplisit menyetel titik akhir untuk Transfer Acceleration ke bucketname .s3-accelerate.amazonaws.com.*

Java

Example

Contoh berikut menunjukkan cara menggunakan titik akhir yang dipercepat untuk mengunggah objek ke Amazon S3. Contoh ini melakukan hal berikut:

- Membuat `AmazonS3Client` yang dikonfigurasi untuk menggunakan titik akhir yang dipercepat. Semua bucket yang diakses klien harus mengaktifkan Transfer Acceleration.
- Memungkinkan Transfer Acceleration pada bucket tertentu. Langkah ini diperlukan hanya jika bucket yang Anda tentukan belum mengaktifkan Transfer Acceleration.
- Memverifikasi bahwa akselerasi transfer diaktifkan untuk bucket tertentu.
- Mengunggah objek baru ke bucket tertentu menggunakan titik akhir akselerasi bucket.

Untuk informasi lebih lanjut tentang Transfer Acceleration, lihat Memulai Amazon S3 Transfer Acceleration. Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketAccelerateConfiguration;
import com.amazonaws.services.s3.model.BucketAccelerateStatus;
import com.amazonaws.services.s3.model.GetBucketAccelerateConfigurationRequest;
import com.amazonaws.services.s3.model.SetBucketAccelerateConfigurationRequest;

public class TransferAcceleration {
    public static void main(String[] args) {
```

```
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ***";

        try {
            // Create an Amazon S3 client that is configured to use the accelerate
 endpoint.
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .enableAccelerateMode()
                    .build();

            // Enable Transfer Acceleration for the specified bucket.
            s3Client.setBucketAccelerateConfiguration(
                    new SetBucketAccelerateConfigurationRequest(bucketName,
                            new BucketAccelerateConfiguration(
                                    BucketAccelerateStatus.Enabled)));

            // Verify that transfer acceleration is enabled for the bucket.
            String accelerateStatus = s3Client.getBucketAccelerateConfiguration(
                    new GetBucketAccelerateConfigurationRequest(bucketName))
                    .getStatus();
            System.out.println("Bucket accelerate status: " + accelerateStatus);

            // Upload a new object using the accelerate endpoint.
            s3Client.putObject(bucketName, keyName, "Test object for transfer
 acceleration");
            System.out.println("Object \"" + keyName + "\" uploaded with transfer
 acceleration.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Contoh berikut menunjukkan cara menggunakan AWS SDK for .NET untuk mengaktifkan Transfer Acceleration pada bucket. Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

Example

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class TransferAccelerationTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            EnableAccelerationAsync().Wait();
        }

        static async Task EnableAccelerationAsync()
        {
                try
                {
                    var putRequest = new PutBucketAccelerateConfigurationRequest
                    {
                        BucketName = bucketName,
                        AccelerateConfiguration = new AccelerateConfiguration
                        {
                            Status = BucketAccelerateStatus.Enabled
                        }
                    };
                    await
 s3Client.PutBucketAccelerateConfigurationAsync(putRequest);
```

```
                    var getRequest = new GetBucketAccelerateConfigurationRequest
                    {
                        BucketName = bucketName
                    };
                    var response = await
 s3Client.GetBucketAccelerateConfigurationAsync(getRequest);

                    Console.WriteLine("Acceleration state = '{0}' ",
 response.Status);
                }
                catch (AmazonS3Exception amazonS3Exception)
                {
                    Console.WriteLine(
                        "Error occurred. Message:'{0}' when setting transfer
 acceleration",
                        amazonS3Exception.Message);
                }
        }
    }
}
```

Saat mengunggah objek ke bucket yang mengaktifkan Transfer Acceleration, Anda menentukan menggunakan titik akhir akselerasi pada saat sedang membuat klien.

```
var client = new AmazonS3Client(new AmazonS3Config
            {
                RegionEndpoint = TestRegionEndpoint,
                UseAccelerateEndpoint = true
            }
```

Javascript

Untuk contoh mengaktifkan Akselerasi Transfer menggunakan AWS SDK for JavaScript, lihat Memanggil operasi putBucketAccelerate Konfigurasi di AWS SDK untuk JavaScript Referensi API.

Python (Boto)

Untuk contoh mengaktifkan Transfer Acceleration dengan menggunakan SDK untuk Python, lihat put_bucket_accelerate_configuration di AWS Referensi API SDK untuk Python (Boto3).

Other

Untuk informasi tentang menggunakan AWS SDK lain, lihat Contoh Kode dan Pustaka.

Penggunaan API REST

Gunakan operasi API REST `PutBucketAccelerateConfiguration` untuk mengaktifkan konfigurasi akselerasi pada bucket yang ada.

Untuk informasi selengkapnya, lihat PutBucketAccelerateConfigurationdi Referensi API Amazon Simple Storage Service.

Menggunakan Alat Perbandingan Kecepatan Amazon S3 Transfer Acceleration

Anda dapat menggunakan Amazon S3 Transfer Acceleration Speed Comparison tool untuk membandingkan kecepatan unggahan yang dipercepat dan tidak dipercepat di seluruh Wilayah Amazon S3. Alat Perbandingan Kecepatan menggunakan unggahan multibagian untuk mentransfer file dari browser Anda ke berbagai Wilayah Amazon S3 dengan dan tanpa menggunakan Akselerasi Transfer.

Anda dapat mengakses Alat Perbandingan Kecepatan menggunakan salah satu metode berikut:

- Salin URL berikut ke jendela browser Anda, ganti *`wilayah`* dengan Wilayah AWS yang Anda gunakan (misalnya,`us-west-2`) dan *`yourBucketName`*dengan nama bucket yang ingin Anda evaluasi:

  ```
  https://s3-accelerate-speedtest.s3-accelerate.amazonaws.com/en/accelerate-speed-comparsion.html?region=region&origBucketName=yourBucketName
  ```

  Untuk daftar titik akhir dan Wilayah Amazon S3, lihat Titik akhir dan kuota Amazon S3 di Referensi Umum AWS.
- Gunakan konsol Amazon S3.

Menggunakan bucket Pembayaran Pemohon untuk transfer dan penggunaan penyimpanan

Secara umum, pemilik bucket membayar semua biaya penyimpanan dan transfer data Amazon S3 yang terkait dengan bucket mereka. Namun, pemilik bucket dapat mengonfigurasi bucket untuk menjadi bucket Pembayaran Pemohon. Dengan bucket Pembayaran Pemohon, pemohon alih-alih pemilik bucket membayar biaya permintaan dan data unduhan dari bucket. Pemilik bucket selalu membayar biaya penyimpanan data.

Biasanya, Anda mengonfigurasi bucket agar menjadi Pembayaran Pemohon ketika Anda ingin berbagi data, tetapi tidak mengeluarkan biaya yang terkait dengan orang lain yang mengakses data tersebut. Sebagai contoh, Anda dapat menggunakan bucket Pembayaran Pemohon saat menyediakan set data besar, seperti direktori kode zip, data referensi, informasi geospasial, atau data perayapan web.

> **Important**
>
> Jika Anda mengaktifkan Pembayaran Pemohon di bucket, akses anonim ke bucket tersebut tidak diizinkan.

Anda harus mengautentikasi semua permintaan yang melibatkan bucket Pembayaran Pemohon. Autentikasi permintaan memungkinkan Amazon S3 untuk mengidentifikasi dan mengenakan biaya atas penggunaan mereka atas bucket Pembayaran Pemohon.

Ketika pemohon mengambil peran AWS Identity and Access Management (IAM) sebelum membuat permintaan mereka, akun tempat peran tersebut dibebankan untuk permintaan tersebut. Untuk informasi selengkapnya tentang peran IAM, lihat Peran IAM dalam Panduan Pengguna IAM.

Setelah Anda mengonfigurasi bucket untuk menjadi bucket Requester Pays, pemohon harus menunjukkan bahwa mereka memahami bahwa mereka akan dikenakan biaya untuk permintaan dan untuk pengunduhan data. Untuk menunjukkan bahwa mereka menerima tagihan, pemohon harus menyertakan `x-amz-request-payer` sebagai header dalam permintaan API mereka untuk permintaan DELETE, GET, HEAD, POST, dan PUT, atau menambahkan `RequestPayer` parameter dalam permintaan REST mereka. Untuk permintaan CLI, pemohon dapat menggunakan parameter. `--request-payer`

Example — Menggunakan Requester Pays saat menghapus objek

Untuk menggunakan contoh DeleteObjectVersionAPI berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

```
DELETE /Key+?versionId=VersionId HTTP/1.1
Host: Bucket.s3.amazonaws.com
x-amz-mfa: MFA
x-amz-request-payer: RequestPayer
x-amz-bypass-governance-retention: BypassGovernanceRetention
x-amz-expected-bucket-owner: ExpectedBucketOwner
```

Jika pemohon mengembalikan objek dengan menggunakan RestoreObjectAPI, Requester Pays didukung selama `x-amz-request-payer` header atau `RequestPayer` parameter ada dalam permintaan; namun, pemohon hanya membayar biaya permintaan. Pemilik ember membayar biaya pengambilan.

Bucket Pembayaran Pemohon tidak mendukung hal berikut:

- Permintaan anonim
- Permintaan SOAP
- Menggunakan bucket Pembayaran Pemohon sebagai bucket tujuan untuk pencatatan log pengguna akhir, atau sebaliknya. Namun, Anda dapat mengaktifkan pembuatan log pengguna akhir pada bucket Pembayaran Pemohon jika bucket targetnya bukan merupakan bucket Pembayaran Pemohon.

Cara kerja Pembayaran Pemohon

Biaya untuk permintaan Pembayaran Pemohon yang berhasil sangatlah sederhana: pemohon membayar transfer data dan permintaan tersebut; pemilik bucket membayar penyimpanan datanya. Namun, pemilik bucket dibebankan untuk permintaan berdasarkan ketentuan berikut:

- Permintaan mengembalikan kesalahan `AccessDenied` (HTTP`403 Forbidden`) dan permintaan dimulai di dalam AWS akun atau AWS organisasi individu pemilik bucket.
- Permintaan tersebut adalah permintaan SOAP.

Untuk informasi selengkapnya tentang Pembayaran Pemohon, lihat topik berikut.

Topik

- Mengonfigurasi Pembayaran Pemohon pada bucket
- Mengambil konfigurasi requestPayment menggunakan API REST
- Mengunduh objek dari ember Requester Pays

Mengonfigurasi Pembayaran Pemohon pada bucket

Anda mengonfigurasikan bucket Amazon S3 untuk menjadi bucket Pembayaran Pemohon, sehingga pemohon membayar biaya permintaan dan pengunduhan data alih-alih pemilik bucket yang melakukannya.

Bagian ini menyediakan contoh bagaimana mengonfigurasi Pembayaran Pemohon pada bucket Amazon S3 menggunakan konsol tersebut dan API REST.

Menggunakan konsol S3

Untuk mengaktifkan Pembayaran Pemohon bucket S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam Bucket, pilih nama bucket yang ingin Anda aktifkan Pembayaran Pemohonnya.
3. Pilih Properti.
4. Pada Pembayaran Pemohon, pilih Edit.
5. Pilih Aktifkan, dan pilih Simpan perubahan.

   Amazon S3 memungkinkan Pembayaran Pemohon untuk bucket Anda, dan menampilkan Ikhtisar bucket. Pada Pembayaran Pemohon, Anda melihat Diaktifkan.

Penggunaan API REST

Hanya pemilik bucket yang dapat mengatur nilai konfigurasi `RequestPaymentConfiguration.payer` dari bucket menjadi `BucketOwner` (default) atau `Requester`. Mengatur sumber daya `requestPayment` bersifat opsional. Secara default, bucket bukan merupakan bucket Pembayaran Pemohon.

Untuk mengembalikan bucket Pembayaran Pemohon ke bucket biasa, Anda menggunakan nilai `BucketOwner`. Biasanya, Anda akan menggunakan `BucketOwner` ketika mengunggah data

ke bucket Amazon S3, dan kemudian Anda akan menetapkan nilai untuk `Requester` sebelum menerbitkan objek di dalam bucket.

Untuk mengatur requestPayment

- Gunakan permintaan PUT untuk mengatur nilai `Payer` ke `Requester` pada bucket tertentu.

```
PUT ?requestPayment HTTP/1.1
Host: [BucketName].s3.amazonaws.com
Content-Length: 173
Date: Wed, 01 Mar 2009 12:00:00 GMT
Authorization: AWS [Signature]

<RequestPaymentConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<Payer>Requester</Payer>
</RequestPaymentConfiguration>
```

Jika permintaannya berhasil, Amazon S3 mengembalikan tanggapan yang serupa dengan yang berikut ini.

```
HTTP/1.1 200 OK
x-amz-id-2: [id]
x-amz-request-id: [request_id]
Date: Wed, 01 Mar 2009 12:00:00 GMT
Content-Length: 0
Connection: close
Server: AmazonS3
x-amz-request-charged:requester
```

Anda dapat mengatur Pembayaran Pemohon hanya pada tingkatan bucket. Anda tidak dapat mengatur Pembayaran Pemohon untuk objek tertentu di dalam bucket.

Anda dapat mengonfigurasi bucket agar menjadi `BucketOwner` atau `Requester` kapan pun. Namun, mungkin beberapa menit dibutuhkan sebelum nilai konfigurasi barunya mulai berlaku.

> **Note**
>
> Pemilik bucket yang memberikan URL yang telah ditentukan sebelumnya harus mempertimbangkan dengan cermat sebelum mengonfigurasi bucket menjadi Pembayaran Pemohon, terutama jika URL tersebut memiliki masa pakai yang lama. Pemilik bucket

dibebankan setiap kali pemohon menggunakan URL yang telah ditandatangani sebelumnya yang menggunakan kredensial pemilik bucket.

Mengambil konfigurasi requestPayment menggunakan API REST

Anda dapat menentukan nilai `Payer` yang ditetapkan di bucket dengan meminta sumber daya `requestPayment`.

Untuk mengembalikan sumber daya requestPayment

- Gunakan permintaan GET untuk mendapatkan sumber daya `requestPayment`, seperti yang ditunjukkan pada permintaan berikut ini.

```
GET ?requestPayment HTTP/1.1
Host: [BucketName].s3.amazonaws.com
Date: Wed, 01 Mar 2009 12:00:00 GMT
Authorization: AWS [Signature]
```

Jika permintaannya berhasil, Amazon S3 mengembalikan tanggapan yang serupa dengan yang berikut ini.

```
HTTP/1.1 200 OK
x-amz-id-2: [id]
x-amz-request-id: [request_id]
Date: Wed, 01 Mar 2009 12:00:00 GMT
Content-Type: [type]
Content-Length: [length]
Connection: close
Server: AmazonS3

<?xml version="1.0" encoding="UTF-8"?>
<RequestPaymentConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<Payer>Requester</Payer>
</RequestPaymentConfiguration>
```

Jawaban ini menunjukkan bahwa nilai `payer` ditetapkan menjadi `Requester`.

Mengunduh objek dari ember Requester Pays

Karena pemohon dikenai biaya untuk mengunduh data melalui bucket Pembayaran Pemohon, permintaan harus mengandung parameter kustom, `x-amz-request-payer`, yang mengonfirmasi bahwa pemohon mengetahui bahwa mereka akan dikenakan biaya untuk pengunduhan tersebut. Untuk mengakses objek di bucket Pembayaran Pemohon, permintaan harus mencakup salah satu dari hal berikut.

- Untuk permintaan DELETE, GET, HEAD, POST, dan PUT, sertakan `x-amz-request-payer : requester` di header
- Untuk URL yang ditandatangani, sertakan `x-amz-request-payer=requester` dalam permintaannya

Jika permintaan berhasil dan pemohon dikenai biaya, responsnya menyertakan header `x-amz-request-charged:requester`. Jika `x-amz-request-payer` tidak ada dalam permintaannya, Amazon S3 mengembalikan kesalahan 403 dan menagih pemilik bucket untuk permintaan tersebut.

> **Note**
>
> Pemilik bucket tidak perlu menambahkan `x-amz-request-payer` atas permintaan mereka. Pastikan bahwa Anda telah menyertakan `x-amz-request-payer` dan nilainya dalam perhitungan tanda tangan Anda. Untuk informasi selengkapnya, lihat Membangun CanonicalizedAmzHeaders Elemen.

Penggunaan API REST

Untuk mengunduh objek dari bucket Pembayaran Pemohon

- Gunakan permintaan `GET` untuk mengunduh sebuah objek dari bucket Pemohon Membayar, seperti yang ditunjukkan dalam permintaan berikut.

```
GET / [destinationObject] HTTP/1.1
Host: [BucketName].s3.amazonaws.com
x-amz-request-payer : requester
Date: Wed, 01 Mar 2009 12:00:00 GMT
Authorization: AWS [Signature]
```

Jika permintaan GET berhasil dan pemohon dikenai biaya, responsnya menyertakan `x-amz-request-charged:requester`.

Amazon S3 dapat mengembalikan kesalahan `Access Denied` pada permintaan yang mencoba untuk mendapatkan objek dari bucket Pembayaran Pemohon. Untuk informasi selengkapnya, lihat Respons Kesalahan dalam Referensi API Amazon Simple Storage Service.

Menggunakan AWS CLI

Untuk mengunduh objek dari bucket Requester Pays menggunakan AWS CLI, Anda tentukan `--request-payer requester` sebagai bagian dari `get-object` permintaan Anda. Untuk informasi selengkapnya, lihat get-object dalam Referensi AWS CLI .

Pembatasan dan batasan bucket

Bucket Amazon S3 dimiliki oleh Akun AWS yang membuatnya. Kepemilikan bucket tidak dapat dialihkan ke akun lain.

Saat Anda membuat ember, Anda memilih namanya dan Wilayah AWS untuk membuatnya. Setelah membuat bucket, Anda tidak dapat mengubah nama atau Wilayahnya.

Saat menamai bucket, Anda harus memilih nama yang relevan bagi Anda atau bisnis Anda. Hindari menggunakan nama yang terkait dengan orang lain. Misalnya, Anda harus menghindari penggunaan `AWS` atau `Amazon` untuk nama bucket Anda.

Secara default, Anda dapat membuat hingga 100 ember di masing-masing ember Anda Akun AWS. Jika memerlukan bucket tambahan, Anda dapat meningkatkan kuota bucket akun hingga maksimal 1.000 bucket dengan mengajukan permintaan penambahan kuota. Tidak ada perbedaan performa baik ketika Anda menggunakan banyak bucket, atau hanya beberapa bucket.

> **Note**
>
> Anda tidak perlu mengirimkan beberapa permintaan peningkatan kuota untuk masing-masing Wilayah AWS. Kuota bucket Anda diterapkan ke Akun AWS Anda.

Untuk informasi tentang cara meningkatkan kuota bucket Anda, buka AWS kuota layanan dalam AWS Referensi Umum.

Menggunakan ulang nama bucket

Jika sebuah bucket kosong, Anda dapat menghapusnya. Setelah Anda menghapus bucket, namanya akan tersedia untuk digunakan kembali. Namun, setelah menghapus bucket, Anda mungkin tidak dapat menggunakan kembali nama tersebut karena berbagai alasan.

Misalnya, ketika Anda menghapus bucket dan namanya menjadi tersedia untuk digunakan kembali, Akun AWS lain mungkin dapat membuat bucket dengan nama tersebut. Selain itu, waktu mungkin telah berlalu sebelum Anda dapat kembali menggunakan nama bucket yang telah dihapus. Jika Anda ingin menggunakan nama bucket yang sama, kami sarankan agar Anda tidak menghapus bucket tersebut.

Untuk informasi selengkapnya tentang bucket, lihat Peraturan penamaan bucket.

Objek dan batasan bucket

Tidak ada ukuran bucket maksimal atau batas jumlah objek yang dapat Anda simpan dalam satu bucket. Anda dapat menyimpan semua objek Anda dalam satu bucket, atau Anda dapat mengaturnya di dalam beberapa bucket. Namun, Anda tidak dapat membuat bucket dari bucket lain.

Operasi bucket

Rekayasa ketersediaan tinggi Amazon S3 berfokus pada operasi get, put, list, dan delete. Karena operasi bucket bekerja terhadap ruang sumber daya global terpusat, tidak disarankan untuk membuat, menghapus, atau mengonfigurasi bucket pada jalur kode ketersediaan tinggi aplikasi Anda. Lebih baik membuat, menghapus, atau mengonfigurasi bucket dalam inisialisasi atau rutinitas pengaturan terpisah yang jarang Anda jalankan.

Penamaan bucket dan bucket yang dibuat secara otomatis

Jika aplikasi Anda membuat bucket secara otomatis, pilih skema penamaan bucket yang kemungkinan kecil akan menimbulkan konflik nama. Pastikan bahwa logika aplikasi Anda akan memilih nama bucket yang berbeda jika nama bucket-nya sudah diambil.

Untuk informasi selengkapnya tentang bucket, lihat Peraturan penamaan bucket.

Mengunggah, mengunduh, dan bekerja dengan objek di Amazon S3

Untuk menyimpan data Anda di Amazon S3, Anda bekerja dengan sumber daya yang dikenal sebagai bucket dan objek. Bucket adalah kontainer untuk objek. Objek adalah file data dan metadata apa pun yang mendeskripsikan file tersebut.

Untuk menyimpan objek di Amazon S3, Anda membuat bucket dan kemudian mengunggah objek tersebut ke dalam bucket. Saat objek berada di dalam bucket, Anda bisa membuka, mengunduh, dan menyalinnya. Bila Anda tidak lagi membutuhkan objek atau bucket, Anda bisa membersihkan sumber daya ini.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

> **Important**
>
> Di dalam konsol Amazon S3, saat Anda memilih Buka atau Unduh Sebagai untuk sebuah objek, operasi ini membuat URL yang telah ditentukan sebelumnya. Selama lima menit, objek Anda akan dapat diakses oleh siapa saja yang memiliki akses ke URL yang telah ditetapkan sebelumnya. Untuk informasi selengkapnya tentang URL yang telah ditandatangani, lihat Menggunakan URLS yang telah ditandatangani.

Dengan Amazon S3, Anda hanya membayar sesuai penggunaan Anda. Untuk informasi selengkapnya tentang fitur dan harga Amazon S3, lihat Amazon S3. Jika Anda adalah pelanggan baru Amazon S3, Anda dapat memulai Amazon S3 secara gratis. Untuk informasi selengkapnya, lihat AWS Tingkat Gratis.

Topik

- Gambaran umum objek Amazon S3
- Membuat nama kunci objek

- Bekerja dengan metadata objek
- Mengunggah Objek
- Mengunggah dan menyalin objek menggunakan unggahan multibagian
- Menyalin, memindahkan, dan mengganti nama objek
- Mengunggah objek
- Memeriksa integritas objek
- Menghapus objek Amazon S3
- Mengatur, membuat daftar, dan bekerja dengan objek Anda
- Bekerja dengan URL yang telah ditandatangani
- Mengubah objek dengan S3 Lambda Objek

Gambaran umum objek Amazon S3

Amazon S3 adalah penyimpanan objek yang menggunakan nilai kunci yang unik untuk menyimpan berapa pun objek yang Anda inginkan. Anda dapat menyimpan objek ini di dalam satu bucket atau lebih, dan setiap objek dapat berukuran hingga 5 TB. Sebuah objek terdiri atas beberapa hal berikut ini:

Kunci

Nama yang Anda tetapkan ke sebuah objek. Anda harus menggunakan kunci objek tersebut untuk mengambil objeknya. Untuk informasi lebih lanjut, lihat Bekerja dengan objek metadata.

ID Versi

Dalam sebuah bucket, kunci dan ID versi akan mengidentifikasikan sebuah objek secara unik. ID versi adalah sebuah string yang dibuat oleh Amazon S3 saat Anda menambahkan sebuah objek ke sebuah bucket. Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.

Nilai

Konten yang Anda simpan.

Sebuah nilai objek dapat berupa urutan byte mana pun. Objek dapat memiliki rentang ukuran yang bervariasi dari nol hingga 5 TB. Untuk informasi selengkapnya, lihat Mengunggah Objek.

Metadata

Seperangkat pasangan nama-nilai yang dapat Anda gunakan untuk menyimpan informasi terkait objek. Anda dapat menetapkan metadata, yang disebut sebagai metadata yang ditentukan pengguna, ke objek Anda dalam Amazon S3. Amazon S3 juga akan menetapkan sistem-metadata ke objek-objek ini, yang akan digunakan untuk mengelola objek. Untuk informasi lebih lanjut, lihat Bekerja dengan objek metadata.

Sub-sumber daya

Amazon S3 menggunakan mekanisme sub-sumber daya untuk menyimpan informasi tambahan terkait objek. Karena sub-sumber daya merupakan subordinat objek, sub-sumber daya selalu dikaitkan dengan beberapa entitas lain seperti objek atau bucket. Untuk informasi selengkapnya, lihat Sub-sumber daya objek.

Informasi kontrol akses

Anda dapat mengontrol akses ke objek yang Anda simpan dalam Amazon S3. Amazon S3 mendukung kontrol akses berbasis sumber daya, seperti daftar kontrol akses (ACL) dan kebijakan bucket, dan kontrol akses berbasis pengguna. Untuk informasi selengkapnya tentang kontrol akses, lihat hal berikut ini:

- Manajemen Akses
- Identity and Access Management untuk Amazon S3
- Mengonfigurasi ACL

Sumber daya Amazon S3 Anda (misalnya, bucket dan objek) secara default bersifat pribadi. Anda harus secara eksplisit memberikan izin kepada orang lain untuk mengakses sumber daya ini. Untuk informasi lebih lanjut tentang objek, lihat Berbagi objek dengan URL yang telah ditandatangani.

Tanda

Anda dapat menggunakan tanda untuk mengategorikan objek yang Anda simpan, untuk mengontrol akses, atau mengalokasikan biaya. Untuk informasi selengkapnya, lihat Mengategorikan penyimpanan Anda menggunakan tag.

Sub-sumber daya objek

Amazon S3 menentukan seperangkat sub-sumber daya yang berkaitan dengan bucket dan objek. Sub-sumber daya merupakan subordinat objek. Artinya, sub-sumber daya tidak dapat muncul

dengan sendirinya. Sub-sumber daya selalu dikaitkan dengan beberapa entitas lain, seperti objek atau bucket.

Tabel berikut ini mencantumkan sub-sumber daya yang terkait dengan objek Amazon S3.

Sub-sumber daya	Deskripsi
acl	Berisi daftar izin yang mengidentifikasi penerima izin serta izin yang diberikan. Saat Anda menciptakan sebuah objek, `acl` mengidentifikasi pemilik objek sebagai pemilik dengan kendali penuh atas objek. Anda dapat mengambil ACL objek atau menggantinya dengan daftar izin yang diperbarui. Setiap pembaruan terhadap ACL mengharuskan Anda untuk mengganti ACL yang ada. Untuk informasi selengkap nya tentang ACL, lihat Gambaran umum daftar kontrol akses (ACL).

Membuat nama kunci objek

Sebuah kunci objek (atau nama kunci) secara unik mengidentifikasi objek dalam sebuah bucket Amazon S3. Metadata objek adalah sekumpulan pasangan nama-nilai. Untuk informasi selengkapnya tentang pemberian metadata objek, lihat Bekerja dengan metadata objek.

Saat Anda membuat objek, Anda menentukan nama kunci, yang secara unik mengidentifikasi objek dalam bucket. Misalnya, di Konsol Amazon S3, jika Anda menyoroti sebuah bucket, daftar objek dalam bucket Anda akan muncul. Nama-nama ini merupakan kunci objek. Nama kunci objek adalah urutan karakter Unicode dengan pengodean UTF-8 hingga 1.024 byte. Nama kunci objek peka terhadap huruf besar atau kecil.

> **Note**
>
> Nama kunci objek dengan nilai “soap” tidak didukung untuk virtual-hosted-style permintaan. Untuk nilai nama kunci objek di mana “soap” digunakan, URL bergaya jalur harus digunakan sebagai gantinya.

Model data Amazon S3 bersifat struktur yang datar: Anda membuat bucket, dan bucket menyimpan objek tersebut. Tidak ada hierarki sub-bucket atau sub-folder. Namun, Anda dapat menyimpulkan

hierarki logis dengan menggunakan prefiks dan pembatas nama kunci, seperti yang dilakukan oleh konsol Amazon S3. Konsol Amazon S3 menggunakan konsep folder. Untuk informasi lebih lanjut tentang cara mengedit metadata dari konsol Amazon S3, lihat Mengedit metadata objek di konsol Amazon S3.

Misalnya, bucket Anda (`admin-created`) memiliki empat objek dengan kunci objek berikut:

```
Development/Projects.xls

Finance/statement1.pdf

Private/taxdocument.pdf

s3-dg.pdf
```

Konsol tersebut menggunakan prefiks nama kunci (`Development/`, `Finance/`, dan `Private/`) serta pembatas ('/') untuk menyajikan struktur folder. Kunci `s3-dg.pdf` tidak memiliki prefiks, sehingga objeknya muncul langsung pada tingkat root bucket. Jika Anda membuka folder `Development/`, Anda melihat objek `Projects.xlsx` di dalamnya.

- Amazon S3 mendukung bucket dan objek, dan tidak memiliki hierarki. Namun, dengan menggunakan awalan dan pembatas dalam nama kunci objek, konsol Amazon S3 dan AWS SDK dapat menyimpulkan hierarki dan memperkenalkan konsep folder.
- Konsol Amazon S3 menerapkan pembuatan folder dengan menciptakan objek nol-byte dengan nilai prefiks dan pembatas folder sebagai kuncinya. Objek folder ini tidak akan muncul dalam konsol. Jika tidak, mereka berperilaku seperti objek lain dan dapat dilihat dan dimanipulasi melalui REST API, AWS CLI, AWS dan SDK.

Panduan penamaan kunci objek

Anda dapat menggunakan karakter UTF-8 apa pun dalam nama kunci objek. Namun, penggunaan karakter tertentu dalam nama kunci dapat menimbulkan masalah pada beberapa aplikasi dan protokol. Panduan berikut ini akan membantu Anda untuk memaksimalkan kepatuhan terhadap DNS, karakter aman web, parser XML, dan API lainnya.

Karakter aman

Set karakter berikut umumnya aman untuk digunakan dalam nama kunci.

Karakter alfanumerik	• 0-9 • a-z • A-Z
Karakter-karakter khusus	• Tanda seru (`!`) • Tanda hubung (`-`) • Garis bawah (_) • Titik (`.`) • Tanda bintang (`*`) • Tanda petik tunggal (`'`) • Tanda kurung buka (`(`) • Tanda kurung tutup (`)`)

Berikut ini adalah contoh nama kunci objek yang valid:

- `4my-organization`
- `my.great_photos-2014/jan/myvacation.jpg`
- `videos/2014/birthday/video1.wmv`

> **Note**
>
> Objek dengan nama kunci yang diakhiri dengan periode "." yang diunduh menggunakan konsol Amazon S3 periode titiknya "." akan dihapus dari nama kunci objek yang diunduh. Untuk mengunduh objek dengan nama kunci yang diakhiri dengan periode “.” yang disimpan di objek yang diunduh, Anda harus menggunakan AWS Command Line Interface (AWS CLI), AWS SDK, atau REST API.
>
> Selain itu, waspadai batasan prefiks berikut ini:
>
> - Objek dengan awalan “./” harus diunggah atau diunduh dengan AWS Command Line Interface (AWS CLI), AWS SDK, atau REST API. Anda tidak bisa menggunakan konsol Amazon S3.
> - Objek dengan prefiks "../" tidak dapat diunggah menggunakan AWS Command Line Interface (AWS CLI) atau konsol Amazon S3.

Karakter yang memerlukan penanganan khusus

Karakter-karakter dalam nama kunci berikut ini mungkin memerlukan penanganan kode tambahan, dan mungkin perlu dikodekan atau direferensikan sebagai HEX. Beberapa dari karakter ini tidak dapat dicetak, dan mungkin tidak dapat ditangani oleh browser Anda, sehingga memerlukan penanganan khusus:

- Ampersand ("&")
- Dolar ("$")
- Karakter ASCII bervariasi dengan rentang hex 00–1F (desimal 0–31) dan 7F (desimal 127)
- Simbol 'At' ("@")
- Tanda sama dengan ("=")
- Titik koma (";")
- Garis miring ("/")
- Titik dua (":")
- Plus ("+")
- Spasi–Urutan spasi yang signifikan dapat dihilangkan dalam beberapa penggunaan (khususnya spasi ganda)
- Tanda koma (",")
- Tanda tanya ("?")

Karakter-karakter yang harus dihindari

Kami menyarankan Anda untuk tidak menggunakan karakter berikut dalam nama kunci karena penanganan karakter khusus yang signifikan, yang tidak konsisten di semua aplikasi.

- Garis miring terbalik ("\")
- Kurung kurawal buka ("{")
- Karakter ASCII yang tidak dapat dicetak (karakter desimal 128–255)
- Tanda pangkat ("^")
- Kurung kurawal tutup ("}")
- Karakter persen ("%")
- Aksen nontirus / back tick ("`")

- Kurung siku tutup ("]")
- Tanda petik
- Simbol 'Lebih Besar Dari' (">")
- Kurung siku buka ("[")
- Tanda ekuivalen ("~")
- Simbol 'Kurang Dari' ("<")
- Karakter 'Pound' ("#")
- Bar vertikal / pipa ("|")

Kendala kunci objek terkait XML

Seperti yang ditentukan oleh standar XHTML pada end-of-line penanganan, semua teks XML-dinormalisasi sedemikian rupa sehingga single carriage return (ASCII code 13) dan carriage return segera diikuti oleh line feed (kode ASCII 10) digantikan oleh karakter feed baris tunggal. Untuk memastikan parsing kunci objek yang benar dalam permintaan XML, pengangkutan kembali dan karakter khusus lainnya harus diganti dengan kode entitas XML yang setara saat dimasukkan ke dalam tag XML. Berikut ini adalah daftar karakter khusus tersebut, serta kode entitas yang setara:

- ' sebagai `'`
- ” sebagai `"`
- & sebagai `&`
- < sebagai `<`
- > sebagai `>`
- \r sebagai `
` atau `
`
- \n sebagai `
` atau `
`

Example

Contoh berikut ini mengilustrasikan penggunaan kode entitas XML sebagai pengganti untuk pengangkutan kembali. Permintaan `DeleteObjects` ini menghapus sebuah objek dengan parameter `key`: `/some/prefix/objectwith\rcarriagereturn` (di mana \r adalah kembali ke awal).

```
<Delete xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
```

```
  <Object>
    <Key>/some/prefix/objectwith&#13;carriagereturn</Key>
  </Object>
</Delete>
```

Bekerja dengan metadata objek

Anda dapat mengatur metadata objek di Amazon S3 pada saat Anda mengunggah objek tersebut. Metadata objek adalah sekumpulan pasangan nama-nilai. Setelah mengunggah objek, Anda tidak dapat memodifikasi metadata objek. Satu-satunya cara untuk memodifikasi metadata objek adalah membuat salinan objek, dan mengatur metadatanya.

Saat Anda membuat objek, Anda juga menentukan nama kunci, yang secara unik mengidentifikasi objek dalam bucket. Sebuah kunci objek (atau nama kunci) secara unik mengidentifikasi objek dalam sebuah bucket Amazon S3. Untuk informasi selengkapnya, lihat Membuat nama kunci objek.

Terdapat dua jenis metadata di Amazon S3: metadata yang ditentukan sistem, dan metadata yang ditentukan pengguna. Bagian berikut ini menyediakan informasi lebih lanjut tentang metadata yang ditentukan sistem dan yang ditentukan pengguna. Untuk informasi lebih lanjut tentang mengedit metadata menggunakan konsol Amazon S3, lihat Mengedit metadata objek di konsol Amazon S3.

Metadata objek yang ditentukan sistem

Untuk setiap objek yang disimpan dalam sebuah bucket, Amazon S3 akan menyimpan serangkaian metadata sistem. Amazon S3 memproses metadata sistem ini sesuai dengan kebutuhan. Misalnya, Amazon S3 mempertahankan metadata tanggal dan ukuran pembuatan objek, dan menggunakan informasi ini sebagai bagian dari manajemen objek.

Terdapat dua kategori metadata sistem:

- Sistem dikontrol–Metadata seperti tanggal pembuatan objek dikontrol oleh sistem, artinya hanya Amazon S3 yang dapat memodifikasi nilai tersebut.
- Dikontrol pengguna–Metadata sistem lainnya, seperti kelas penyimpanan yang dikonfigurasi untuk objek dan apakah enkripsi di sisi server objek diaktifkan, adalah beberapa contoh metadata sistem yang nilai-nilainya Anda kendalikan. Jika bucket Anda dikonfigurasi sebagai situs web, terkadang Anda mungkin ingin mengalihkan permintaan halaman ke halaman lainnya, atau ke URL eksternal. Dalam hal ini, halaman laman web adalah sebuah objek di dalam bucket Anda. Amazon S3 menyimpan nilai pengalihan halaman sebagai metadata sistem yang nilainya Anda kendalikan.

Saat membuat objek, Anda dapat mengonfigurasi nilai item metadata sistem ini, atau memperbarui nilai saat Anda membutuhkannya. Untuk informasi selengkapnya tentang kelas penyimpanan, lihat Menggunakan kelas penyimpanan Amazon S3.

Amazon S3 menggunakan AWS KMS kunci untuk mengenkripsi objek Amazon S3 Anda. AWS KMS mengenkripsi hanya data objek. Checksum, bersama dengan algoritma yang ditentukan, disimpan sebagai bagian dari metadata objek. Jika enkripsi di sisi server diminta untuk objek, maka checksum disimpan dalam bentuk terenkripsi. Untuk informasi lebih lanjut tentang enkripsi di sisi server, lihat Melindungi data dengan enkripsi.

> **Note**
>
> Header permintaan PUT dibatasi hingga ukuran 8 KB. Dalam header permintaan PUT, metadata yang ditentukan sistem dibatasi ke dalam ukuran 2 KB. Ukuran metadata yang ditentukan sistem diukur dengan menjumlahkan jumlah byte dalam pengodean US-ASCII dari setiap kunci dan nilai.

Tabel berikut ini menyediakan daftar metadata yang ditentukan sistem, dan apakah Anda dapat memperbaruinya.

Nama	Deskripsi	Apakah pengguna dapat mengubah nilainya?
`Date`	Tanggal dan waktu saat ini.	Tidak
`Cache-Control`	Bidang header umum yang digunakan untuk menentukan kebijakan caching.	Ya
`Content-Disposition`	Informasi presentasi objek.	Ya
`Content-Length`	Ukuran objek dalam byte.	Tidak

Nama	Deskripsi	Apakah pengguna dapat mengubah nilainya?
`Content-Type`	Jenis objek.	Ya
`Last-Modified`	Tanggal pembuatan objek atau tanggal modifikasi terakhir, mana pun yang terbaru. Untuk unggahan multibagian, tanggal pembuatan objek adalah tanggal inisiasi unggahan multibagian.	Tidak
`ETag`	Tag entitas (ETag) yang mewakili versi objek tertentu. Untuk objek yang tidak diunggah sebagai unggahan multibagian dan tidak terenkripsi atau dienkripsi oleh enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3), ETag adalah intisari dari data MD5.	Tidak
`x-amz-server-side-encryption`	Header yang menunjukkan apakah enkripsi sisi server diaktifkan untuk objek, dan apakah enkripsi tersebut menggunakan kunci AWS Key Management Service (AWS KMS) (SSE-KMS) atau menggunakan kunci enkripsi terkelola Amazon S3 (SSE-S3). Untuk informasi selengkapnya, lihat Melindungi data dengan enkripsi di sisi klien.	Ya
`x-amz-checksum-crc32` , `x-amz-checksum-crc32c` , `x-amz-checksum-sha1` , `x-amz-checksum-sha256`	Header yang berisi checksum atau intisari objek. Biasanya, salah satu header ini akan diatur pada satu waktu, bergantung pada algoritma checksum yang Anda perintahkan untuk digunakan oleh Amazon S3. Untuk informasi selengkapnya tentang memilih algoritma checksum, lihat Memeriksa integritas objek.	Tidak

Nama	Deskripsi	Apakah pengguna dapat mengubah nilainya?
`x-amz-version-id`	Versi objek tersebut. Saat Anda mengaktifkan Penentuan Versi pada bucket, Amazon S3 menetapkan ID versi ke objek yang ditambahkan ke bucket. Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.	Tidak
`x-amz-delete-marker`	Penanda Boolean yang mengindikasikan apakah objek tersebut adalah penanda hapus. Penanda ini hanya digunakan dalam bucket yang mengaktifkan Penentuan Versi,	Tidak
`x-amz-storage-class`	Kelas penyimpanan yang digunakan untuk menyimpan objek tersebut. Untuk informasi selengkapnya, lihat Menggunakan kelas penyimpanan Amazon S3.	Ya
`x-amz-website-redirect-location`	Sebuah header yang mengalihkan permintaan objek terkait ke objek lain di bucket yang sama, atau ke URL eksternal. Untuk informasi selengkapnya, lihat (Opsional) Mengonfigurasi pengalihan halaman web.	Ya
`x-amz-server-side-encryption-aws-kms-key-id`	Header yang menunjukkan ID kunci KMS enkripsi AWS KMS simetris yang digunakan untuk mengenkripsi objek. Header ini hanya digunakan ketika header `x-amz-server-side-encryption` muncul, dan memiliki nilai `aws:kms`.	Ya
`x-amz-server-side-encryption-customer-algorithm`	Header yang menunjukkan apakah enkripsi di sisi server dengan kunci enkripsi yang disediakan pelanggan (SSE-C) telah diaktifkan. Untuk informasi selengkapnya, lihat Menggunakan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C).	Ya

Nama	Deskripsi	Apakah pengguna dapat mengubah nilainya?
`x-amz-tagging`	Tag-set untuk objek tersebut. Kumpulan tag harus dikodekan sebagai parameter Kueri URL.	Ya

Metadata objek yang ditentukan pengguna

Saat mengunggah objek, Anda juga dapat menetapkan metadata pada objek tersebut. Anda memberikan informasi opsional ini sebagai pasangan nama-nilai (nilai-kunci) saat Anda mengirimkan permintaan `PUT` atau `POST` untuk membuat objek. Saat Anda mengunggah objek menggunakan API REST, nama metadata opsional yang ditentukan pengguna harus dimulai dengan `x-amz-meta-` untuk membedakannya dari header HTTP lainnya. Saat Anda mengambil objek menggunakan API REST, prefiks ini akan ditampilkan. Saat Anda mengunggah objek menggunakan SOAP API, prefiks tidak dibutuhkan. Saat Anda mengambil objek menggunakan SOAP API, prefiks akan dihapus, tanpa menghiraukan API apa yang Anda gunakan untuk mengunggah objek tersebut.

> **Note**
>
> Dukungan SOAP melalui HTTP tidak digunakan lagi, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami sarankan Anda menggunakan REST API atau AWS SDK.

Saat metadata diambil melalui API REST, Amazon S3 menggabungkan header yang memiliki nama yang sama (kasus pengabaian) ke dalam daftar yang dibatasi koma. Jika beberapa metadata berisi karakter yang tidak dapat dicetak, maka metadata tidak akan ditampilkan. Sebaliknya, header `x-amz-missing-meta` ditampilkan dengan nilai jumlah dari entri metadata yang tidak dapat dicetak. Tindakan `HeadObject` mengambil metadata dari sebuah objek tanpa mengembalikan objek itu sendiri. Operasi ini berguna jika Anda hanya tertarik pada metadata objek. Untuk menggunakan `HEAD`, Anda harus memiliki akses `READ` ke objek. Untuk informasi selengkapnya, lihat HeadObjectdi Referensi API Amazon Simple Storage Service.

Metadata yang ditentukan pengguna adalah sekumpulan pasangan nilai-kunci. Amazon S3 menyimpan kunci metadata yang ditentukan pengguna dalam huruf kecil.

Amazon S3 memungkinkan karakter Unicode arbitrer dalam nilai metadata Anda.

Untuk menghindari masalah seputar penyajian nilai metadata ini, Anda harus menyesuaikan penggunaan karakter US-ASCII saat menggunakan REST, dan UTF-8 saat menggunakan SOAP atau unggahan berbasis browser melalui `POST`.

Saat menggunakan karakter non US-ASCII dalam nilai metadata Anda, string Unicode yang disediakan untuk karakter non US-ASCII akan diperiksa. Nilai header tersebut didekodekan karakternya sesuai RFC 2047 sebelum disimpan dan dikodekan sesuai RFC 2047 agar aman untuk email sebelum dikembalikan. Jika string hanya berisi karakter US-ASCII, string akan ditampilkan sebagaimana adanya.

Berikut adalah contohnya.

```
PUT /Key HTTP/1.1
Host: DOC-EXAMPLE-BUCKET1.s3.amazonaws.com
x-amz-meta-nonascii: ÄMÄZÕÑ S3

HEAD /Key HTTP/1.1
Host: DOC-EXAMPLE-BUCKET1.s3.amazonaws.com
x-amz-meta-nonascii: =?UTF-8?B?w4PChE3Dg8KEWsODwpXDg8KRIFMz?=

PUT /Key HTTP/1.1
Host: DOC-EXAMPLE-BUCKET1.s3.amazonaws.com
x-amz-meta-ascii: AMAZONS3

HEAD /Key HTTP/1.1
Host: DOC-EXAMPLE-BUCKET1.s3.amazonaws.com
x-amz-meta-ascii: AMAZONS3
```

> **Note**
>
> Header permintaan PUT dibatasi hingga ukuran 8 KB. Dalam header permintaan PUT, metadata yang ditentukan pengguna dibatasi hingga ke dalam ukuran 2 KB. Ukuran metadata yang ditentukan pengguna diukur dengan menjumlahkan jumlah byte dalam pengodean UTF-8 dari setiap kunci dan nilai.

Untuk informasi tentang mengubah metadata objek Anda setelah diunggah dengan membuat salinan objek, memodifikasinya, dan mengganti objek lama, atau membuat versi baru, lihat Mengedit metadata objek di konsol Amazon S3.

Mengedit metadata objek di konsol Amazon S3

Anda dapat menggunakan konsol Amazon S3 untuk mengedit metadata Objek S3 yang sudah ada. Beberapa metadata diatur oleh Amazon S3 saat Anda mengunggah objek. Misalnya, `Content-Length` dan `Last-Modified` merupakan bidang metadata objek yang ditentukan sistem yang tidak dapat dimodifikasi oleh pengguna.

Anda juga dapat mengatur beberapa metadata saat mengunggah objek, dan mengeditnya nanti saat kebutuhan Anda berubah. Misalnya, Anda mungkin memiliki serangkaian objek yang awalnya Anda simpan di dalam kelas penyimpanan `STANDARD`. Seiring waktu, Anda mungkin tidak lagi membutuhkan ketersediaan data ini. Sehingga, Anda mengubah kelas penyimpanan untuk `GLACIER` dengan mengedit nilai dari kunci `x-amz-storage-class` dari `STANDARD` ke `GLACIER`.

> **Note**
>
> Pertimbangkan masalah berikut saat Anda mengedit objek metadata di Amazon S3:
>
> - Tindakan ini membuat salinan objek dengan pengaturan yang diperbarui dan tanggal modifikasi terakhir. Jika Penentuan Versi S3 diaktifkan, versi baru objek akan dibuat, dan objek yang sudah ada menjadi versi yang lebih lama. Jika Penentuan Versi S3 tidak diaktifkan, salinan baru dari objek menggantikan objek asli. Akun AWS Terkait dengan peran IAM yang mengubah properti juga menjadi pemilik objek baru atau (versi objek).
> - Untuk menggunakan konsol Amazon S3 untuk mengedit metadata untuk objek yang memiliki tag yang ditentukan pengguna, Anda juga harus memiliki izin. `s3:GetObjectTagging` Jika Anda menggunakan konsol Amazon S3 untuk mengedit metadata untuk objek yang tidak memiliki tag yang ditentukan pengguna tetapi berukuran lebih dari 16 MB, Anda juga harus memiliki izin. `s3:GetObjectTagging`
>
> Jika kebijakan bucket tujuan menolak `s3:GetObjectTagging` tindakan, metadata untuk objek akan diperbarui, tetapi tag yang ditentukan pengguna akan dihapus dari objek, dan Anda akan menerima kesalahan.
> - Mengedit nilai pembaruan metadata untuk nama kunci yang sudah ada.

- Objek yang dienkripsi dengan kunci enkripsi yang disediakan pelanggan (SSE-C) tidak dapat disalin menggunakan konsol tersebut. Anda harus menggunakan AWS CLI, AWS SDK, atau Amazon S3 REST API.

> **Warning**
>
> Saat mengedit metadata folder, tunggu agar operasi `Edit metadata` selesai sebelum menambahkan objek baru ke folder. Jika tidak, objek baru juga akan diedit.

Topik berikut ini menjelaskan cara untuk mengedit metadata objek menggunakan konsol Amazon S3.

Mengedit metadata yang ditentukan sistem

Anda dapat mengonfigurasi beberapa, namun tidak semua, metadata sistem untuk Objek S3. Untuk daftar metadata yang ditentukan sistem dan apakah Anda dapat mengubah nilainya, lihat Metadata objek yang ditentukan sistem.

Untuk mengedit metadata objek yang ditentukan sistem

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Navigasi ke bucket Amazon S3 atau folder Anda, dan pilih kotak centang di sebelah kiri nama objek dengan metadata yang ingin Anda edit.
3. Pada menu Tindakan, pilih Edit tindakan, dan pilih Edit metadata.
4. Tinjau objek yang tercantum, lalu pilih Tambahkan metadata.
5. Untuk metadata Jenis, pilih Ditentukan sistem.
6. Tentukan Kunci unik dan Nilai metadata.
7. Untuk mengedit metadata tambahan, pilih Tambahkan metadata. Anda juga dapat memilih Hapus untuk menghapus satu set type-key-values.
8. Setelah selesai, pilih Edit metadata dan Amazon S3 akan mengedit metadata objek yang ditentukan.

Mengedit metadata yang ditentukan pengguna

Anda dapat mengedit metadata objek yang ditentukan pengguna dengan menggabungkan prefiks metadata, `x-amz-meta-`, dan nama yang Anda pilih untuk membuat kunci kustom. Misalnya, jika Anda menambahkan nama kustom `alt-name`, kunci metadata adalah `x-amz-meta-alt-name`.

Metadata yang ditentukan pengguna dapat berukuran sebesar 2 KB. Untuk menghitung ukuran total metadata yang ditentukan pengguna, hitung jumlah byte dalam pengodean UTF-8 untuk setiap kunci dan nilai. Kunci dan nilainya harus sesuai dengan standar US-ASCII. Untuk informasi selengkapnya, lihat Metadata objek yang ditentukan pengguna.

Untuk mengedit metadata objek yang ditentukan pengguna

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang berisi objek yang ingin Anda tambahkan metadatanya.

   Anda juga dapat pergi ke folder.
3. Di daftar Objek, pilih kotak centang di samping nama objek yang ingin Anda tambahkan metadatanya.
4. Di menu Tindakan, pilih Edit metadata.
5. Tinjau objek yang tercantum, lalu pilih Tambahkan metadata.
6. Untuk metadata Jenis, pilih Ditetapkan pengguna.
7. Memasukkan Kunci kustom yang unik, yang mengikuti `x-amz-meta-`. Masukkan juga metadata Nilai.
8. Untuk menambahkan metadata tambahan, pilih Tambahkan metadata. Anda juga dapat memilih Hapus untuk menghapus satu set type-key-values.
9. Pilih Edit metadata.

   Amazon S3 mengedit metadata objek yang ditentukan.

Mengunggah Objek

Saat Anda mengunggah file ke Amazon S3, file tersebut disimpan sebagai objek S3. Objek terdiri dari data file dan metadata yang menjelaskan objek tersebut. Anda dapat memiliki jumlah objek tak terbatas dalam satu bucket. Sebelum Anda dapat mengunggah file ke bucket Amazon S3, Anda

memerlukan izin menulis untuk bucket tersebut. Untuk informasi selengkapnya tentang izin akses, lihat Identity and Access Management untuk Amazon S3.

Anda dapat mengunggah jenis file apa pun—gambar, cadangan, data, film, dan sebagainya—ke dalam bucket S3. Ukuran maksimum file yang dapat Anda unggah menggunakan konsol Amazon S3 adalah 160 GB. Untuk mengunggah file yang lebih besar dari 160 GB, gunakan AWS Command Line Interface (AWS CLI), AWS SDK, atau Amazon S3 REST API.

Jika Anda mengunggah sebuah objek dengan nama kunci yang sudah ada dalam bucket yang diaktifkan dengan Penentuan Versi, Amazon S3 membuat versi lain dari objek tersebut dan bukan mengganti objek yang ada. Untuk informasi selengkapnya tentang penentuan versi, lihat Menggunakan konsol S3.

Tergantung dari ukuran data yang Anda unggah, Amazon S3 menawarkan opsi berikut:

- Unggah objek dalam satu operasi dengan menggunakan AWS SDK, REST API, atau AWS CLI — Dengan satu PUT operasi, Anda dapat mengunggah satu objek berukuran hingga 5 GB.
- Unggah objek tunggal menggunakan konsol Amazon S3 –Dengan konsol Amazon S3, Anda dapat mengunggah objek tunggal yang berukuran hingga 160 GB.
- Unggah objek dalam beberapa bagian dengan menggunakan AWS SDK, REST API, atau AWS CLI — Menggunakan operasi API unggahan multibagian, Anda dapat mengunggah satu objek besar, berukuran hingga 5 TB.

  Pengoperasian API unggahan multibagian dirancang untuk meningkatkan pengalaman pengunggahan untuk objek yang lebih besar. Anda dapat mengunggah objek dalam beberapa bagian. Bagian-bagian objek ini dapat diunggah secara mandiri, dalam urutan apa pun, dan secara paralel. Anda dapat menggunakan unggahan multibagian untuk objek dengan ukuran mulai dari 5 MB hingga 5 TB. Untuk informasi selengkapnya, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.

Saat Anda mengunggah sebuah objek, objek secara otomatis dienkripsi menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) secara default. Saat Anda mengunduhnya, objek tersebut didekripsi. Untuk informasi selengkapnya, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3 dan Melindungi data dengan enkripsi.

Saat mengunggah objek, jika Anda ingin menggunakan jenis enkripsi default yang berbeda, Anda juga dapat menentukan enkripsi di sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS) dalam permintaan PUT S3, atau mengatur konfigurasi enkripsi default di bucket

tujuan untuk menggunakan SSE-KMS untuk mengenkripsi data Anda. Untuk informasi lebih lanjut tentang SSE-KMS, lihat Menentukan enkripsi di sisi server dengan AWS KMS (SSE-KMS). Jika Anda ingin menggunakan kunci KMS yang dimiliki oleh akun lain, Anda harus memiliki izin untuk menggunakan kunci tersebut. Untuk informasi selengkapnya tentang izin lintas akun untuk kunci KMS, lihat Membuat kunci KMS yang dapat digunakan oleh akun lain di Panduan Pengembang AWS Key Management Service .

Jika Anda menemukan error Access Denied (403 Forbidden) di Amazon S3, Pecahkan masalah kesalahan Akses Ditolak (403 Forbidden) di Amazon S3 lihat untuk mempelajari lebih lanjut tentang penyebab umumnya.

Menggunakan konsol S3

Prosedur ini menjelaskan cara untuk mengunggah objek dan folder ke bucket Amazon S3 menggunakan konsol tersebut.

Ketika Anda mengunggah objek, nama kunci objek adalah nama file dan prefiks opsional. Di konsol Amazon S3, Anda dapat membuat folder untuk mengatur objek Anda. Di Amazon S3, folder diwakili sebagai prefiks yang muncul dalam nama kunci objek. Jika Anda mengunggah objek individu ke folder di konsol Amazon S3, nama folder disertakan dalam nama kunci objek.

Misalnya, jika Anda mengunggah sebuah objek bernama `sample1.jpg` ke folder bernama `backup`, nama kuncinya adalah `backup/sample1.jpg`. Namun, objek ditampilkan di konsol sebagai `sample1.jpg` dalam folder `backup`. Untuk informasi selengkapnya tentang nama kunci, lihat Bekerja dengan metadata objek.

> **Note**
>
> Jika Anda mengganti sebuah objek atau mengubah properti apa pun di konsol Amazon S3, misalnya Kelas Penyimpanan, Enkripsi, atau Metadata, sebuah objek baru dibuat untuk menggantikan yang lama. Jika Penentuan Versi S3 diaktifkan, versi baru objek akan dibuat, dan objek yang sudah ada menjadi versi yang lebih lama. Peran yang mengubah properti juga menjadi pemilik objek baru (atau versi objek).

Saat Anda mengunggah folder, Amazon S3 mengunggah semua file dan subfolder dari folder yang ditentukan ke bucket Anda. Kode tersebut kemudian menetapkan nama kunci objek yang merupakan kombinasi nama file unggahan dan nama folder. Misalnya, jika Anda mengunggah folder bernama `/images` yang berisi dua file, `sample1.jpg` dan `sample2.jpg`, Amazon S3 mengunggah

file tersebut dan kemudian menetapkan nama kunci yang sesuai, `images/sample1.jpg` dan `images/sample2.jpg`. Nama-nama kunci termasuk nama folder sebagai prefiks. Konsol Amazon S3 hanya menampilkan bagian dari nama kunci yang mengikuti `/` yang terakhir. Misalnya, dalam folder `images`, objek `images/sample1.jpg` dan `images/sample2.jpg` ditampilkan sebagai `sample1.jpg` dan `sample2.jpg`.

Untuk mengunggah file dan folder ke bucket S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di dalam daftar Bucket, pilih nama bucket tempat Anda ingin mengunggah folder dan file.
4. Pilih Unggah.
5. Di jendela Unggah, lakukan salah satu hal berikut:
   - Seret dan jatuhkan file dan folder ke jendela Unggah.
   - Pilih Tambahkan file atau Tambahkan folder, pilih file atau folder untuk diunggah, dan pilih Buka.
6. Untuk mengaktifkan Penentuan Versi, di bawah Tujuan memilih Aktifkan Penentuan Versi Bucket.
7. Untuk mengunggah file dan folder yang terdaftar tanpa mengonfigurasi opsi pengunggahan tambahan, di bagian bawah halaman, pilih Unggah.

   Amazon S3 mengunggah objek dan folder Anda. Setelah unggahan selesai, Anda melihat pesan sukses di halaman Unggahan: status.

Untuk mengonfigurasi properti objek tambahan

1. Untuk mengubah izin daftar kontrol akses, pilih Izin.
2. Di bagian bawah Daftar kontrol akses (ACL), edit izinnya.

   Untuk informasi selengkapnya tentang izin akses objek, lihat Menggunakan konsol S3 untuk mengatur izin ACL untuk objek. Anda dapat memberikan akses baca ke objek Anda kepada publik (semua orang di dunia) untuk semua file yang Anda unggah. Namun, kami merekomendasikan untuk tidak mengubah pengaturan default untuk mengakses baca publik. Pemberian akses baca publik berlaku untuk sebagian kecil kasus penggunaan, seperti saat

bucket digunakan untuk situs web. Anda dapat selalu mengubah izin objek setelah Anda mengunggah objek.

3. Untuk mengonfigurasi properti tambahan lainnya, pilih Properti.
4. Di bagian bawah Kelas penyimpanan pilih kelas penyimpanan untuk file yang Anda unggah.

   Untuk informasi selengkapnya tentang kelas penyimpanan, lihat Menggunakan kelas penyimpanan Amazon S3.
5. Untuk memperbarui pengaturan enkripsi untuk objek Anda, di bagian bawah Pengaturan enkripsi di sisi server, lakukan hal berikut.

   a. Pilih Tentukan kunci enkripsi.

   b. Di bawah Pengaturan enkripsi, pilih Gunakan pengaturan bucket untuk enkripsi default, atau Ganti pengaturan bucket untuk enkripsi default.

   c. Jika Anda memilih Ganti pengaturan bucket untuk enkripsi default, Anda harus mengonfigurasi pengaturan enkripsi berikut.

      - Untuk mengenkripsi file yang diunggah dengan menggunakan kunci yang dikelola oleh Amazon S3, pilih Kunci yang dikelola Amazon S3 (SSE-S3).

        Untuk informasi selengkapnya, lihat Menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3).
      - Untuk mengenkripsi file yang diunggah dengan menggunakan kunci yang disimpan di AWS Key Management Service (AWS KMS), pilih AWS Key Management Service kunci (SSE-KMS). Lalu pilih salah satu opsi berikut ini untuk kunci AWS KMS :
        - Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari AWS KMS keys Anda, dan pilih Kunci KMS Anda dari daftar kunci yang tersedia.

          Kunci Kunci yang dikelola AWS (`aws/s3`) dan kunci terkelola pelanggan Anda muncul dalam daftar ini. Untuk informasi selengkapnya tentang CMK, lihat Kunci pelanggan dan AWS kunci di AWS Key Management Service Panduan Pengembang.
        - Untuk memasukkan ARN kunci KMS, pilih Masukkan AWS KMS key ARN, lalu masukkan ARN kunci KMS Anda di bidang yang muncul.
        - Untuk membuat kunci terkelola pelanggan baru di AWS KMS konsol, pilih Buat kunci KMS.

          Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat kunci di Panduan AWS Key Management Service Pengembang.

> **Important**
>
> Anda hanya dapat menggunakan tombol KMS yang tersedia Wilayah AWS sama dengan bucket. Konsol Amazon S3 hanya mencantumkan kunci 100 KMS pertama di Wilayah yang sama dengan bucket. Untuk menggunakan kunci KMS yang tidak terdaftar, Anda harus memasukkan ARN kunci KMS Anda. Jika Anda ingin menggunakan kunci KMS yang dimiliki oleh akun lain, Anda harus terlebih dahulu memiliki izin untuk menggunakan kunci tersebut, dan kemudian Anda harus memasukkan ARN kunci KMS.
> Amazon S3 hanya mendukung kunci KMS enkripsi simetris, dan tidak mendukung kunci KMS asimetris. Untuk informasi selengkapnya, lihat Mengidentifikasi tombol KMS simetris dan asimetris dalam Panduan Pengembang AWS Key Management Service .

6. Untuk menggunakan checksum tambahan, pilih Aktif. Kemudian untuk Fungsi checksum, pilih fungsi yang ingin Anda gunakan. Amazon S3 menghitung dan menyimpan nilai checksum setelah menerima objek secara keseluruhan. Anda dapat menggunakan kotak Nilai prakalkulasi untuk memberikan nilai yang telah dihitung sebelumnya. Jika sudah, Amazon S3 membandingkan nilai yang Anda berikan dengan nilai yang dihitung. Jika kedua nilai tidak cocok, Amazon S3 menghasilkan kesalahan.

   Checksum tambahan memungkinkan Anda menentukan algoritma checksum yang ingin Anda gunakan untuk memverifikasi data Anda. Untuk informasi selengkapnya tentang checksum tambahan, lihat Memeriksa integritas objek.

7. Untuk menambahkan tanda ke semua objek yang Anda unggah, pilih Tambahkan tag. Masukkan nama tag di bidang Kunci. Masukkan nilai untuk tanda.

   Penandaan objek memberi Anda cara untuk mengategorikan penyimpanan. Setiap tag adalah pasangan nilai kunci. Kunci dan nilai tag peka huruf besar dan kecil. Anda dapat membuat hingga 10 tanda per objek. Kunci tanda dapat terdiri dari hingga 128 karakter Unicode, dan nilai tanda dapat terdiri dari hingga 256 karakter Unicode. Untuk informasi selengkapnya tentang tag objek, lihat Mengategorikan penyimpanan Anda menggunakan tag.

8. Untuk menambahkan metadata, pilih Tambahkan metadata.

   a. Di bawah Jenis, pilih Ditentukan sistem atau Ditentukan pengguna.

Untuk metadata yang ditentukan sistem, Anda dapat memilih header HTTP umum, seperti Jenis Konten dan Konten-Disposisi. Untuk daftar metadata yang ditentukan sistem, serta informasi tentang apakah Anda dapat menambahkan nilai, lihat Metadata objek yang ditentukan sistem. Metadata apa pun yang dimulai dengan prefiks `x-amz-meta-` diperlakukan sebagai metadata yang ditentukan pengguna. Metadata yang ditentukan pengguna disimpan bersama objek, dan dikembalikan saat Anda mengunduh objek tersebut. Baik kunci maupun nilainya harus sesuai dengan standar US-ASCII. Metadata yang ditentukan pengguna dapat berukuran sebesar 2 KB. Untuk informasi selengkapnya tentang metadata yang ditentukan sistem dan yang ditentukan pengguna, lihat Bekerja dengan metadata objek.

b. Untuk Kunci, pilih kuncinya.

c. Ketikkan nilai untuk kunci tersebut.

9. Untuk mengunggah objek Anda, pilih Unggah.

   Amazon S3 mengunggah objek Anda. Setelah unggahannya selesai, Anda dapat melihat pesan sukses di halaman Unggah: status.

10. Pilih Keluar.

Menggunakan AWS SDK

Anda dapat menggunakan AWS SDK untuk mengunggah objek di Amazon S3. SDK menyediakan pustaka wrapper agar Anda dapat mengunggah data dengan mudah. Untuk informasi lebih lanjut, lihat Daftar SDK yang didukung.

Berikut adalah beberapa contoh dengan beberapa pilihan SDK:

.NET

Contoh kode C# berikut ini membuat dua objek dengan dua permintaan `PutObjectRequest`:

- Permintaan `PutObjectRequest` yang pertama menyimpan sebuah string teks sebagai data objek sampel. Permintaan ini juga menentukan nama kunci bucket dan objek.
- Permintaan `PutObjectRequest` yang kedua mengunggah sebuah file dengan menentukan nama file tersebut. Permintaan ini juga menentukan header `ContentType` dan metadata objek opsional (sebuah judul).

Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class UploadObjectTest
    {
        private const string bucketName = "*** bucket name ***";
        // For simplicity the example creates two objects from the same file.
        // You specify key names for these objects.
        private const string keyName1 = "*** key name for first object created ***";
        private const string keyName2 = "*** key name for second object created
 ***";
        private const string filePath = @"*** file path ***";
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.EUWest1;

        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            WritingAnObjectAsync().Wait();
        }

        static async Task WritingAnObjectAsync()
        {
            try
            {
                // 1. Put object-specify only key name for the new object.
                var putRequest1 = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName1,
                    ContentBody = "sample text"
                };
```

```
                PutObjectResponse response1 = await
 client.PutObjectAsync(putRequest1);

                // 2. Put the object-set ContentType and add metadata.
                var putRequest2 = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName2,
                    FilePath = filePath,
                    ContentType = "text/plain"
                };

                putRequest2.Metadata.Add("x-amz-meta-title", "someTitle");
                PutObjectResponse response2 = await
 client.PutObjectAsync(putRequest2);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine(
                        "Error encountered ***. Message:'{0}' when writing an
 object"
                        , e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine(
                    "Unknown encountered on server. Message:'{0}' when writing an
 object"
                    , e.Message);
            }
        }
    }
}
```

Java

Contoh berikut ini menampilkan pembuatan dua objek. Objek pertama memiliki string teks sebagai data, dan objek kedua adalah sebuah file. Contoh ini membuat objek pertama dengan menentukan nama bucket, kunci objek, dan data teks dalam panggilan langsung ke `AmazonS3Client.putObject()`. Contoh ini membuat objek kedua dengan menggunakan `PutObjectRequest` yang menentukan nama bucket, kunci objek, dan jalur file. `PutObjectRequest` juga menentukan header `ContentType` dan metadata judul.

Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectMetadata;
import com.amazonaws.services.s3.model.PutObjectRequest;

import java.io.File;
import java.io.IOException;

public class UploadObject {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String stringObjKeyName = "*** String object key name ***";
        String fileObjKeyName = "*** File object key name ***";
        String fileName = "*** Path to file to upload ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .build();

            // Upload a text string as a new object.
            s3Client.putObject(bucketName, stringObjKeyName, "Uploaded String
 Object");

            // Upload a file as a new object with ContentType and title specified.
            PutObjectRequest request = new PutObjectRequest(bucketName,
 fileObjKeyName, new File(fileName));
            ObjectMetadata metadata = new ObjectMetadata();
            metadata.setContentType("plain/text");
            metadata.addUserMetadata("title", "someTitle");
            request.setMetadata(metadata);
```

```
            s3Client.putObject(request);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

JavaScript

Contoh berikut ini menampilkan pengunggahan sebuah file yang ada ke bucket Amazon S3 di Wilayah tertentu.

```
import { PutObjectCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new PutObjectCommand({
    Bucket: "test-bucket",
    Key: "hello-s3.txt",
    Body: "Hello S3!",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

PHP

Contoh ini memandu Anda menggunakan kelas dari AWS SDK for PHP untuk mengunggah objek berukuran hingga 5 GB. Untuk file yang lebih besar, Anda harus menggunakan operasi

pengunggahan API multibagian. Untuk informasi selengkapnya, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.

Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

Example — Membuat sebuah objek dalam sebuah bucket Amazon S3 dengan mengunggah data

Contoh PHP berikut ini membuat objek dalam bucket tertentu dengan mengunggah data menggunakan metode `putObject()`.

```
 require 'vendor/autoload.php';

use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

try {
    // Upload data.
    $result = $s3->putObject([
        'Bucket' => $bucket,
        'Key'    => $keyname,
        'Body'   => 'Hello, world!',
        'ACL'    => 'public-read'
    ]);

    // Print the URL to the object.
    echo $result['ObjectURL'] . PHP_EOL;
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Ruby

The AWS SDK for Ruby - Versi 3 memiliki dua cara untuk mengunggah objek ke Amazon S3. Yang pertama menggunakan pengunggah file terkelola, yang mempermudah pengunggahan file berukuran berapa pun dari disk. Untuk menggunakan metode pengunggah file terkelola:

1. Ciptakan contoh dari kelas `Aws::S3::Resource`.
2. Referensikan objek target berdasarkan nama dan kunci bucket. Objek muncul di dalam bucket dan memiliki kunci unik yang mengidentifikasi setiap objek.
3. Panggil`#upload_file` pada objek.

Example

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectUploadFileWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Uploads a file to an Amazon S3 object by using a managed uploader.
  #
  # @param file_path [String] The path to the file to upload.
  # @return [Boolean] True when the file is uploaded; otherwise false.
  def upload_file(file_path)
    @object.upload_file(file_path)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't upload file #{file_path} to #{@object.key}. Here's why:
 #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
```

```
  object_key = "my-uploaded-file"
  file_path = "object_upload_file.rb"

  wrapper = ObjectUploadFileWrapper.new(Aws::S3::Object.new(bucket_name,
 object_key))
  return unless wrapper.upload_file(file_path)

  puts "File #{file_path} successfully uploaded to #{bucket_name}:#{object_key}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Cara kedua bahwa AWS SDK for Ruby - Versi 3 dapat meng-upload objek menggunakan `#put` metode`Aws::S3::Object`. Hal ini berguna jika objeknya adalah string atau objek I/O yang bukan merupakan file pada disk. Untuk menggunakan metode ini:

1. Buat instans dari kelas `Aws::S3::Resource`.
2. Referensikan objek target berdasarkan nama dan kunci bucket.
3. Panggil`#put`, pindahkan ke dalam string atau objek I/O.

Example

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectPutWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  def put_object(source_file_path)
    File.open(source_file_path, "rb") do |file|
      @object.put(body: file)
    end
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't put #{source_file_path} to #{object.key}. Here's why:
 #{e.message}"
```

```
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object-key"
  file_path = "my-local-file.txt"

  wrapper = ObjectPutWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
  success = wrapper.put_object(file_path)
  return unless success

  puts "Put file #{file_path} into #{object_key} in #{bucket_name}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Penggunaan API REST

Anda dapat mengirim permintaan REST untuk mengunggah sebuah objek. Anda dapat mengirim permintaan PUT untuk mengunggah dalam satu operasi. Untuk informasi lebih lanjut, lihat PUT Objek.

Menggunakan AWS CLI

Anda dapat mengirim permintaan PUT untuk mengunggah objek hingga 5 GB dalam satu operasi. Untuk informasi dan contoh selengkapnya, lihat contoh `PutObject` di AWS CLI Referensi Perintah.

Mengunggah dan menyalin objek menggunakan unggahan multibagian

Unggahan multibagian memungkinkan Anda untuk mengunggah satu objek sebagai kumpulan bagian. Setiap bagian merupakan bagian data objek yang saling berkaitan. Anda dapat mengunggah bagian-bagian objek tersebut secara independen dan dengan urutan apa pun. Jika ada transmisi bagian mana pun yang gagal, Anda dapat mentransmisikan ulang bagian tersebut tanpa memengaruhi bagian lainnya. Setelah semua bagian objek Anda diunggah, Amazon S3 merakit bagian-bagian ini dan menciptakan objek. Secara umum, saat ukuran objek Anda mencapai 100 MB,

Anda harus mempertimbangkan untuk menggunakan unggahan multibagian daripada mengunggah objek tersebut dalam satu operasi.

Penggunaan unggahan multibagian memberikan keuntungan sebagai berikut:

- Peningkatan throughput–Anda dapat mengunggah bagian-bagian secara paralel untuk meningkatkan throughput.
- Pemulihan yang cepat dari masalah jaringan apa pun–Ukuran bagian yang lebih kecil meminimalkan dampak pengunggahan ulang karena kesalahan jaringan.
- Jeda dan pelanjutan pengunggahan objek–Anda dapat mengunggah bagian-bagian objek kapan saja. Setelah Anda memulai unggahan multibagian, tidak ada kedaluwarsa; Anda harus secara eksplisit menyelesaikan atau menghentikan unggahan multibagian.
- Mulai unggahan sebelum Anda mengetahui ukuran akhir objek–Anda dapat mengunggah sebuah objek selagi Anda membuatnya.

Kami menyarankan agar Anda menggunakan unggahan multibagian dengan cara berikut:

- Jika Anda mengunggah objek besar melalui jaringan dengan bandwidth tinggi yang stabil, gunakan unggahan multibagian untuk memaksimalkan penggunaan bandwidth yang tersedia dengan mengunggah bagian-bagian objek secara paralel untuk performa multi-thread.
- Jika Anda mengunggah melalui jaringan yang tidak teratur, gunakan unggahan multibagian untuk meningkatkan ketahanan terhadap kesalahan jaringan dengan menghindari pengunggahan ulang. Saat menggunakan unggahan multibagian, Anda harus mencoba mengunggah lagi bagian-bagian yang terganggu selama pengunggahan. Anda tidak perlu mengunggah ulang objek Anda dari awal.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori. Untuk informasi selengkapnya tentang penggunaan unggahan multibagian dengan S3 Express One Zone dan bucket direktori, lihat Menggunakan unggahan multibagian dengan bucket direktori.

Proses pengunggahan multibagian

Pengunggahan multibagian merupakan proses tiga langkah: Anda memulai pengunggahan, Anda mengunggah bagian objek, dan setelah Anda mengunggah semua bagiannya, Anda menyelesaikan pengunggahan multibagian tersebut. Setelah menerima permintaan unggahan multibagian yang selesai, Amazon S3 membangun objek dari bagian-bagian yang diunggah, dan kemudian Anda dapat mengakses objek tersebut sebagaimana Anda ingin mengakses objek lain di dalam bucket Anda.

Anda dapat membuat daftar semua unggahan multibagian yang sedang berlangsung, atau mendapatkan daftar bagian yang telah Anda unggah untuk unggahan multibagian tertentu. Setiap operasi ini dijelaskan dalam bagian ini.

Menginisiasi unggahan multibagian

Saat Anda mengirim permintaan untuk memulai unggahan multibagian, Amazon S3 mengirimkan respons dengan ID unggahan, yang merupakan pengidentifikasi unik untuk unggahan multibagian Anda. Anda harus menyertakan ID unggahan ini setiap kali Anda mengunggah bagian, mendaftar bagian, menyelesaikan unggahan, atau menghentikan sebuah unggahan. Jika Anda ingin memberikan metadata apa pun yang menjelaskan objek yang sedang diunggah, Anda harus menyediakannya dalam permintaan untuk memulai unggahan multibagian.

Pengunggahan bagian-bagian

Saat mengunggah sebuah bagian, selain ID pengunggahan, Anda harus menentukan nomor bagiannya. Anda dapat memilih nomor bagian antara 1 hingga 10.000. Nomor bagian secara unik mengidentifikasi sebuah bagian dan posisinya dalam objek yang Anda unggah. Nomor bagian yang Anda pilih tidak harus berurutan (misalnya, nomornya dapat berupa 1, 5, dan 14). Jika Anda mengunggah sebuah bagian baru menggunakan nomor yang sama dengan bagian yang diunggah sebelumnya, bagian yang diunggah sebelumnya akan ditimpa.

Saat Anda mengunggah bagian, Amazon S3 mengembalikan tag entitas (ETag) untuk bagian tersebut sebagai header dalam respons. Untuk setiap unggah bagian, Anda harus mencatat nomor bagian dan nilai ETag. Anda harus memasukkan nilai-nilai ini dalam permintaan selanjutnya untuk menyelesaikan unggahan multipart. Setiap bagian akan memiliki ETag sendiri pada saat diunggah. Namun, setelah unggahan multipart selesai dan semua bagian dikonsolidasikan, semua bagian akan berada di bawah satu ETag sebagai checksum checksum.

> **Note**
>
> Setelah Anda memulai pengunggahan multibagian dan mengunggah satu atau beberapa bagian, Anda harus menyelesaikan atau menghentikan pengunggahan multibagian agar tidak dikenakan biaya penyimpanan dari bagian yang diunggah. Hanya setelah Anda menyelesaikan atau menghentikan unggahan multibagian, Amazon S3 akan mengosongkan penyimpanan suku cadang dan berhenti menagih Anda untuk penyimpanan suku cadang. Setelah menghentikan unggahan multibagian, Anda tidak dapat mengunggah bagian apa pun yang menggunakan ID unggahan tersebut lagi. Jika terdapat unggahan bagian yang sedang berlangsung, bagian tersebut masih bisa berhasil atau gagal bahkan setelah Anda menghentikan pengunggahan. Untuk memastikan Anda mengosongkan semua penyimpanan yang digunakan oleh semua bagian, Anda harus menghentikan unggahan multibagian hanya setelah semua unggahan bagian selesai.

Penyelesaian unggahan multibagian

Saat Anda menyelesaikan unggahan multibagian, Amazon S3 membuat objek dengan menggabungkan bagian-bagian dalam urutan menaik berdasarkan nomor bagian. Jika metadata objek disediakan dalam permintaan mulai unggahan multibagian, Amazon S3 akan mengaitkan metadatanya dengan objek tersebut. Setelah permintaan selesai sepenuhnya, bagiantersebut tidak akan ada lagi.

Permintaan penyelesaian unggahan multibagian Anda harus menyertakan ID unggahan dan daftar nomor bagian maupun nilai ETag yang sesuai. Respons Amazon S3 mencakup sebuah ETag yang secara unik mengidentifikasi data objek gabungan. ETag ini belum tentu merupakan hash MD5 dari data objek.

Contoh panggilan unggahan multibagian

Untuk contoh ini, asumsikan bahwa Anda membuat unggahan multibagian untuk file yang berukuran 100 GB. Dalam hal ini, Anda akan memiliki panggilan API berikut untuk seluruh proses. Akan ada total 1002 panggilan API.

- Panggilan `CreateMultipartUpload` untuk memulai proses.
- Panggilan `UploadPart` 1000 individu, masing-masing mengunggah sebagian dari 100 MB, untuk ukuran total 100 GB.
- Panggilan `CompleteMultipartUpload` untuk menyelesaikan proses.

Pendaftaran unggahan multibagian

Anda dapat mendaftar bagian-bagian dari unggahan multibagian tertentu atau semua unggahan multibagian yang sedang berlangsung. Operasi daftar bagian menampilkan informasi bagian yang telah Anda unggah untuk unggahan multibagian tertentu. Untuk setiap permintaan daftar bagian, Amazon S3 akan menampilkan informasi bagian untuk unggahan multibagian tertentu, hingga maksimum 1.000 bagian. Jika ada lebih dari 1.000 bagian dalam unggahan multibagian, Anda harus mengirim serangkaian permintaan daftar bagian untuk mengambil semua bagian. Perhatikan bahwa daftar bagian yang ditampilkan tidak mencakup bagian yang belum selesai diunggah. Dengan menggunakan operasi daftar unggahan multibagian, Anda dapat memperoleh daftar unggahan multibagian yang sedang berlangsung.

Unggahan multibagian yang sedang berlangsung adalah unggahan yang telah Anda mulai, tetapi belum selesai atau dihentikan. Setiap permintaan akan ditampilkan sebanyak maksimum 1.000 unggahan multibagian. Jika ada lebih dari 1.000 unggahan multibagian yang sedang berlangsung, Anda harus mengirim permintaan tambahan untuk mengambil unggahan multibagian yang tersisa. Gunakan pendaftaran yang ditampilkan untuk verifikasi. Jangan menggunakan hasil pendaftaran ini saat mengirim permintaan selesaikan unggahan multibagian. Sebaliknya, simpan daftar nomor bagian Anda sendiri yang Anda tentukan saat mengunggah bagian dan nilai ETag yang ditampilkan oleh Amazon S3.

Checksum dengan operasi unggahan multibagian

Saat mengunggah sebuah objek ke Amazon S3, Anda dapat menentukan algoritma checksum untuk Amazon S3 untuk digunakan. Amazon S3 menggunakan MD5 secara default untuk memverifikasi integritas data; namun, Anda dapat menentukan algoritma checksum tambahan yang akan digunakan. Saat menggunakan MD5, Amazon S3 menghitung checksum dari seluruh objek multibagian setelah unggahannya selesai. Checksum ini bukan merupakan checksum dari seluruh objek, melainkan checksum untuk setiap bagian individu.

Saat Anda menginstruksikan Amazon S3 untuk menggunakan checksum tambahan, Amazon S3 menghitung nilai checksum untuk setiap bagian dan menyimpan nilainya. Anda dapat menggunakan API atau SDK untuk mengambil nilai checksum untuk masing-masing bagian dengan menggunakan `GetObject` atau `HeadObject`. Jika Anda ingin mengambil nilai checksum untuk masing-masing bagian dari unggahan multibagian yang masih dalam proses, Anda dapat menggunakan `ListParts`.

> **Important**
>
> Jika Anda menggunakan unggahan multibagian dengan checksum tambahan, nomor bagian multibagian harus menggunakan nomor bagian berturut-turut. Saat menggunakan checksum tambahan, jika Anda mencoba menyelesaikan permintaan unggahan multibagian dengan nomor bagian yang tidak berurutan, Amazon S3 akan membuat kesalahan `500 Internal Server Error` HTTP.

Untuk informasi selengkapnya tentang cara kerja checksum dengan objek multibagian, lihat Memeriksa integritas objek.

Operasi pengunggahan multibagian serentak

Dalam lingkungan pengembangan terdistribusi, aplikasi Anda dapat memulai beberapa pembaruan pada objek yang sama secara bersamaan. Aplikasi Anda dapat memulai beberapa unggahan multibagian menggunakan kunci objek yang sama. Untuk setiap unggahan ini, aplikasi Anda kemudian dapat mengunggah bagian dan mengirim sebuah permintaan menyelesaikan unggahan ke Amazon S3 untuk membuat objek. Jika Penentuan Versi S3 diaktifkan pada bucket, menyelesaikan pengunggahan multibagian akan selalu membuat versi baru. Untuk bucket dengan Penentuan Versi yang tidak diaktifkan, ada kemungkinan bahwa permintaan lain yang diterima antara saat unggahan multibagian dimulai dan diselesaikan akan didahulukan.

> **Note**
>
> Ada kemungkinan permintaan lain yang diterima antara saat Anda memulai unggahan multibagian dan menyelesaikannya akan didahulukan. Misalnya, jika operasi lain menghapus kunci setelah Anda memulai unggahan multibagian dengan kunci tersebut, tetapi sebelum Anda menyelesaikannya, keseluruhan respons unggahan multibagian mungkin mengindikasikan sebuah keberhasilan penciptaan objek tanpa Anda dapat melihat objek tersebut.

Unggahan multibagian dan harga

Setelah Anda memulai unggahan multibagian, Amazon S3 akan menyimpan semua bagian hingga Anda menyelesaikan atau menghentikan unggahan. Sepanjang masa pakainya, Anda akan ditagih

untuk semua penyimpanan, bandwidth, dan permintaan untuk unggahan multibagian ini dan bagian terkaitnya.

Bagian-bagian ini dibebankan sesuai dengan kelas penyimpanan yang ditentukan saat bagiannya diunggah. Pengecualian untuk ini adalah bagian yang diunggah ke S3 Glacier Flexible Retrieval, atau S3 Glacier Deep Archive. Bagian multibagian yang sedang berlangsung untuk PUT ke kelas penyimpanan S3 Glacier Flexible Retrieval ditagih sebagai S3 Glacier Flexible Retrieval Staging Storage dengan tarif penyimpanan Standar S3 hingga pengunggahan selesai. Selain itu, keduanya CreateMultipartUpload dan UploadPart ditagih dengan tarif Standar S3. Hanya CompleteMultipartUpload permintaan yang ditagih dengan tarif Pengambilan Fleksibel S3 Glacier. Demikian pula, suku cadang multipart yang sedang berlangsung untuk PUT ke kelas penyimpanan S3 Glacier Deep Archive ditagih sebagai S3 Glacier Flexible Retrieval Staging Staging Storage pada tingkat penyimpanan Standar S3 hingga unggahan selesai, dengan hanya permintaan yang dikenakan tarif S3 Glacier Deep Archive. CompleteMultipartUpload

Jika Anda menghentikan pengunggahan multibagian, Amazon S3 akan menghapus artefak unggahan dan bagian apa pun yang telah Anda unggah, dan Anda tidak lagi ditagih untuk hal tersebut. Tidak ada biaya penghapusan awal untuk menghapus unggahan multibagian yang belum selesai terlepas dari kelas penyimpanan yang ditentukan. Untuk informasi selengkapnya tentang harga, lihat Harga Amazon S3.

> **Note**
>
> Untuk meminimalkan biaya penyimpanan, kami menyarankan untuk Anda mengonfigurasi aturan siklus hidup untuk menghapus unggahan multibagian yang tidak lengkap setelah beberapa hari tertentu dengan menggunakan tindakan `AbortIncompleteMultipartUpload`. Untuk informasi selengkapnya tentang membuat aturan siklus hidup untuk menghapus unggahan multibagian yang tidak lengkap, lihat Mengonfigurasi konfigurasi siklus hidup bucket untuk menghapus unggahan multibagian yang tidak lengkap.

Dukungan API untuk unggahan multibagian

Pustaka ini menyediakan abstraksi tingkat tinggi yang mempermudah pengunggahan objek multibagian. Namun, jika aplikasi Anda membutuhkan, Anda dapat menggunakan API REST secara langsung. Bagian berikut dalam Referensi API Amazon Simple Storage Service menjelaskan tentang API REST untuk unggahan multibagian.

Untuk panduan unggahan multibagian yang menggunakan fungsi AWS Lambda, lihat Mengunggah objek besar ke Amazon S3 menggunakan akselerasi unggahan dan transfer multibagian.

- Buat Unggahan Multibagian
- Unggah Bagian
- Unggah Bagian (Salinan)
- Selesaikan Unggahan Multibagian
- Batalkan Unggahan Multibagian
- Daftarkan Bagian
- Daftarkan Unggahan Multibagian

AWS Command Line Interface dukungan untuk upload multipart

Topik berikut dalam AWS Command Line Interface menjelaskan operasi untuk upload multipart.

- Mulai Unggahan Multibagian
- Unggah Bagian
- Unggah Bagian (Salinan)
- Selesaikan Unggahan Multibagian
- Batalkan Unggahan Multibagian
- Daftarkan Bagian
- Daftarkan Unggahan Multibagian

AWS Dukungan SDK untuk unggahan multipart

Anda dapat menggunakan AWS SDK untuk mengunggah objek di beberapa bagian. Untuk daftar AWS SDK yang didukung oleh tindakan API, lihat:

- Buat Unggahan Multibagian
- Unggah Bagian
- Unggah Bagian (Salinan)
- Selesaikan Unggahan Multibagian
- Batalkan Unggahan Multibagian

- Daftarkan Bagian
- Daftarkan Unggahan Multibagian

API dan izin unggahan multibagian

Anda harus memiliki izin yang diperlukan untuk menggunakan operasi pengunggahan multibagian. Anda dapat menggunakan daftar kontrol akses (ACL), kebijakan bucket, atau kebijakan pengguna untuk memberikan izin kepada individu guna melakukan operasi ini. Tabel berikut mencantumkan izin yang diperlukan untuk berbagai operasi unggahan multibagian saat menggunakan ACL, kebijakan bucket, atau kebijakan pengguna.

Tindakan	Izin yang diperlukan
Buat Unggahan Multibagian	Anda harus diizinkan untuk melakukan tindakan `s3:PutObject` pada sebuah objek untuk membuat unggahan multibagian. Pemilik bucket dapat mengizinkan pengguna utama lain untuk melakukan tindakan `s3:PutObject` .
Mulai Unggahan Multibagian	Anda harus diizinkan untuk melakukan tindakan `s3:PutObject` pada sebuah objek untuk memulai unggahan multibagian. Pemilik bucket dapat mengizinkan pengguna utama lain untuk melakukan tindakan `s3:PutObject` .
Inisiator	Elemen kontainer yang mengidentifikasi siapa yang memulai unggahan multibagian. Jika inisiator adalah Akun AWS, elemen ini memberikan informasi yang sama dengan elemen Pemilik. Jika inisiator adalah seorang pengguna IAM, elemen ini akan menyediakan ARN pengguna dan nama tampilan.
Unggah Bagian	Anda harus diizinkan untuk melakukan tindakan `s3:PutObject` pada sebuah objek untuk mengunggah sebuah bagian. Pemilik bucket harus mengizinkan inisiator untuk melakukan tindakan `s3:PutObject` pada sebuah objek agar inisiator dapat mengunggah bagian untuk objek tersebut.

Tindakan	Izin yang diperlukan
Unggah Bagian (Salinan)	Anda harus diizinkan untuk melakukan tindakan `s3:PutObject` pada sebuah objek untuk mengunggah sebuah bagian. Karena Anda mengunggah sebuah bagian dari objek yang sudah ada, Anda harus diizinkan `s3:GetObject` pada objek sumber. Agar inisiator dapat mengunggah bagian untuk sebuah objek, pemilik bucket harus mengizinkan inisiator untuk melakukan tindakan `s3:PutObject` pada objek.
Selesaikan Unggahan Multibagian	Anda harus diizinkan untuk melakukan tindakan `s3:PutObject` pada sebuah objek untuk menyelesaikan sebuah unggahan multibagian. Pemilik bucket harus mengizinkan inisiator untuk melakukan tindakan `s3:PutObject` pada sebuah objek agar inisiator dapat menyelesaikan sebuah unggahan multibagian untuk objek tersebut.
Hentikan Unggahan Multibagian	Anda harus diizinkan untuk melakukan tindakan `s3:AbortMultipartUpload` untuk menghentikan sebuah unggahan multibagian. Secara default, pemilik bucket dan inisiator unggahan multibagian diizinkan untuk melakukan tindakan ini sebagai bagian dari kebijakan IAM dan bucket. Jika inisiator adalah pengguna IAM, pengguna tersebut juga Akun AWS diizinkan untuk menghentikan unggahan multibagian tersebut. Dengan kebijakan titik akhir VPC, inisiator unggahan multibagian tidak secara otomatis mendapatkan izin untuk melakukan tindakan `s3:AbortMultipartUpload` . Selain pengaturan default ini, pemilik bucket dapat mengizinkan pengguna utama lain untuk melakukan tindakan `s3:AbortMultipartUpload` terhadap suatu objek. Pemilik bucket dapat menolak kemampuan pengguna utama mana pun untuk melakukan tindakan `s3:AbortMultipartUpload` .

Tindakan	Izin yang diperlukan
Daftarkan Bagian	Anda harus diizinkan untuk melakukan tindakan `s3:ListMultipartUploadParts` untuk mendaftar bagian dalam unggahan multibagian. Secara default, pemilik bucket memiliki izin untuk mendaftarkan bagian untuk unggahan multibagian apa pun pada bucket. Inisiator unggahan multibagian memiliki izin untuk mendaftarkan bagian-bagian dari unggahan multibagian tertentu. Jika inisiator unggahan multipart adalah pengguna IAM, Akun AWS pengendali pengguna IAM tersebut juga memiliki izin untuk membuat daftar bagian dari unggahan tersebut. Selain pengaturan default ini, pemilik bucket dapat mengizinkan pengguna utama lain untuk melakukan tindakan `s3:ListMultipartUploadParts` terhadap suatu objek. Pemilik bucket dapat juga menolak kemampuan pengguna utama mana pun untuk melakukan tindakan `s3:ListMultipartUploadParts` .
Daftarkan Unggahan Multibagian	Anda harus diizinkan untuk melakukan tindakan `s3:ListBucketMultipartUploads` pada sebuah bucket untuk mendaftar unggahan multibagian yang sedang berlangsung pada bucket tersebut. Selain pengaturan default ini, pemilik bucket dapat mengizinkan pengguna utama lain untuk melakukan tindakan `s3:ListBucketMultipartUploads` terhadap suatu objek.
AWS KMS Enkripsi dan Dekripsi izin terkait	Untuk melakukan unggahan multipart dengan enkripsi menggunakan kunci AWS Key Management Service (AWS KMS) KMS, pemohon harus memiliki izin untuk `kms:Decrypt` dan `kms:GenerateDataKey` tindakan pada kunci tersebut. Izin ini diperlukan karena Amazon S3 harus mendekripsi dan membaca data dari bagian file terenkripsi sebelum menyelesaikan unggahan multibagian. Untuk informasi lebih lanjut, lihat Mengunggah sebuah file besar ke Amazon S3 dengan enkripsi menggunakan AWS KMS key dalam AWS Pusat Pengetahuan. Jika pengguna atau peran IAM Anda Akun AWS sama dengan kunci KMS, maka Anda harus memiliki izin ini pada kebijakan kunci. Jika pengguna atau peran IAM Anda memiliki akun yang berbeda dengan kunci KMS Anda, maka Anda harus memiliki izin pada kebijakan kunci maupun pengguna atau pengguna IAM Anda.

Untuk informasi tentang hubungan antara izin ACL dan izin dalam kebijakan akses, lihat Pemetaan izin ACL dan izin kebijakan akses. Untuk informasi selengkapnya tentang pengguna IAM dan praktik terbaik, lihat Identitas IAM (pengguna, grup pengguna, dan peran) di Panduan Pengguna IAM.

Topik

- Mengonfigurasi siklus hidup bucket untuk menghapus unggahan multibagian yang tidak lengkap
- Pengunggahan objek menggunakan unggahan multibagian
- Mengunggah direktori menggunakan kelas .NET tingkat tinggi TransferUtility
- Mendaftarkan unggahan multibagian
- Melacak unggahan multibagian
- Membatalkan unggahan multibagian
- Menyalin objek menggunakan unggahan multibagian
- Batas unggahan multibagian Amazon S3

Mengonfigurasi siklus hidup bucket untuk menghapus unggahan multibagian yang tidak lengkap

Sebagai praktik terbaik, kami menyarankan Anda untuk mengonfigurasi aturan siklus hidup dengan menggunakan tindakan `AbortIncompleteMultipartUpload` untuk meminimalkan biaya penyimpanan Anda. Untuk informasi lebih lanjut tentang membatalkan unggahan multibagian, lihat Membatalkan unggahan multibagian.

Amazon S3 mendukung aturan siklus hidup bucket yang dapat Anda gunakan untuk mengarahkan Amazon S3 menghentikan unggahan multibagian yang tidak diselesaikan dalam jumlah hari tertentu setelah prosesnya dimulai. Jika unggahan multibagian tidak selesai dalam jangka waktu yang ditentukan, unggahan tersebut memenuhi syarat untuk operasi pembatalan. Amazon S3 kemudian menghentikan unggahan multibagian dan menghapus bagian yang terkait dengan unggahan multibagian. Aturan ini berlaku untuk unggahan multibagian yang ada dan yang Anda buat nanti.

Berikut ini adalah contoh konfigurasi siklus hidup yang menentukan aturan dengan tindakan `AbortIncompleteMultipartUpload`.

```
<LifecycleConfiguration>
    <Rule>
        <ID>sample-rule</ID>
        <Prefix></Prefix>
```

```
        <Status>Enabled</Status>
        <AbortIncompleteMultipartUpload>
          <DaysAfterInitiation>7</DaysAfterInitiation>
        </AbortIncompleteMultipartUpload>
    </Rule>
</LifecycleConfiguration>
```

Dalam contoh ini, aturan tidak menentukan nilai untuk elemen `Prefix` (prefiks nama kunci objek). Oleh karena itu, aturan ini berlaku untuk semua objek dalam bucket tempat Anda memulai unggahan multibagian. Setiap unggahan multibagian yang dimulai dan tidak diselesaikan dalam waktu tujuh hari memenuhi syarat untuk operasi pembatalan. Tindakan membatalkan tidak memengaruhi unggahan multibagian yang sudah selesai.

Untuk informasi selengkapnya tentang konfigurasi siklus hidup bucket, lihat Mengelola siklus hidup penyimpanan Anda.

> **Note**
>
> Jika unggahan multibagian selesai dalam jumlah hari yang ditentukan dalam aturan, tindakan siklus hidup `AbortIncompleteMultipartUpload` tidak berlaku lagi (yaitu, Amazon S3 tidak akan mengambil tindakan apa pun). Selain itu, tindakan ini tidak berlaku untuk objek. Tidak ada objek yang dihapus oleh tindakan siklus hidup ini. Selain itu, Anda tidak akan dikenakan biaya penghapusan awal untuk Siklus Hidup S3 saat Anda menghapus bagian unggahan multibagian yang tidak lengkap.

Menggunakan konsol S3

Untuk mengelola unggahan multibagian yang tidak lengkap secara otomatis, Anda dapat menggunakan konsol S3 untuk membuat aturan siklus hidup untuk mengakhiri byte unggahan multibagian yang tidak lengkap dari bucket Anda setelah jumlah hari yang ditentukan. Prosedur berikut menunjukkan cara menambahkan aturan siklus hidup untuk menghapus unggahan multibagian yang tidak lengkap setelah 7 hari. Untuk informasi selengkapnya tentang cara menambahkan aturan siklus hidup, lihat Menyetel konfigurasi siklus hidup pada bucket.

Untuk menambahkan aturan siklus hidup untuk membatalkan unggahan multibagian yang tidak lengkap yang berumur lebih dari 7 hari

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di daftar Bucket, pilih nama bucket yang ingin Anda buatkan aturan siklus hidupnya.
3. Pilih tab Manajemen, dan pilih Buat aturan siklus hidup.
4. Di Nama aturan siklus hidup, masukkan nama untuk aturan Anda.

   Nama dalam bucket harus unik.
5. Pilih cakupan aturan siklus hidup:

   - Untuk membuat aturan siklus hidup untuk semua objek dengan prefiks tertentu, pilih Batasi cakupan aturan ini menggunakan satu atau lebih filter, dan masukkan bidang Prefiks.
   - Untuk membuat aturan siklus hidup ini pada semua objek dalam bucket, pilih Aturan ini berlaku untuk semua objek di dalam bucket, lalu pilih Saya menyatakan bahwa aturan ini berlaku untuk semua objek dalam bucket.
6. Di bagian bawah Tindakan aturan siklus hidup, pilih Hapus penanda hapus objek kedaluwarsa atau unggahan multibagian yang tidak lengkap.
7. Di bagian bawah Hapus penanda hapus objek kedaluwarsa atau unggahan multibagian yang tidak lengkap, pilih Hapus unggahan multibagian yang tidak lengkap.
8. Di bidang Jumlah hari, masukkan jumlah hari setelahnya untuk menghapus unggahan multibagian yang tidak lengkap (untuk contoh ini, 7 hari).
9. Pilih Buat aturan.

Menggunakan AWS CLI

Perintah berikut `put-bucket-lifecycle-configuration`AWS Command Line Interface (AWS CLI) menambahkan konfigurasi siklus hidup untuk bucket yang ditentukan. Untuk menggunakan perintah ini, ganti *`user input placeholders`* dengan informasi Anda.

```
aws s3api put-bucket-lifecycle-configuration  \
        --bucket example-s3-bucket1  \
        --lifecycle-configuration filename-containing-lifecycle-configuration
```

Contoh berikut menunjukkan cara menambahkan aturan siklus hidup untuk membatalkan unggahan multibagian yang tidak lengkap dengan menggunakan AWS CLI. Ini mencakup contoh konfigurasi siklus hidup JSON untuk membatalkan unggahan multibagian yang tidak lengkap yang berusia lebih dari 7 hari.

Untuk menggunakan perintah CLI dalam contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

Untuk menambahkan aturan siklus hidup untuk membatalkan unggahan multibagian yang tidak lengkap

1. Mengatur AWS CLI. Untuk petunjuk, lihat Mengembangkan dengan Amazon S3 menggunakan AWS CLI.
2. Simpan konfigurasi siklus hidup contoh berikut dalam sebuah file (misalnya, *lifecycle.json*). Konfigurasi contoh ini menentukan prefiks kosong, dan karenanya berlaku untuk semua objek di bucket. Untuk membatasi konfigurasi ke subset objek, Anda dapat menentukan prefiks.

```
{
    "Rules": [
        {
            "ID": "Test Rule",
            "Status": "Enabled",
            "Filter": {
                "Prefix": ""
            },
            "AbortIncompleteMultipartUpload": {
                "DaysAfterInitiation": 7
            }
        }
    ]
}
```

3. Jalankan perintah CLI berikut ini untuk menetapkan konfigurasi siklus hidup pada bucket Anda.

```
aws s3api put-bucket-lifecycle-configuration   \
--bucket example-s3-bucket1  \
--lifecycle-configuration file://lifecycle.json
```

4. Untuk memverifikasi bahwa konfigurasi siklus hidup telah ditetapkan pada bucket Anda, ambil konfigurasi siklus hidup dengan menggunakan perintah `get-bucket-lifecycle` berikut.

```
aws s3api get-bucket-lifecycle  \
--bucket example-s3-bucket1
```

5. Untuk menghapus konfigurasi siklus hidup, gunakan perintah `delete-bucket-lifecycle` berikut ini.

```
aws s3api delete-bucket-lifecycle \
--bucket example-s3-bucket1
```

Pengunggahan objek menggunakan unggahan multibagian

Anda dapat menggunakan unggahan multibagian untuk mengunggah satu objek ke Amazon S3 secara terprogram.

Untuk informasi selengkapnya, silakan lihat bagian-bagian berikut ini.

Menggunakan AWS SDK (API tingkat tinggi)

Beberapa AWS SDK mengekspos API tingkat tinggi yang menyederhanakan pengunggahan multibagian dengan menggabungkan berbagai operasi API yang diperlukan untuk menyelesaikan unggahan multibagian ke dalam satu operasi. Untuk informasi selengkapnya, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.

Jika Anda perlu menjeda dan melanjutkan unggahan multipart, memvariasikan ukuran bagian selama pengunggahan, atau tidak mengetahui ukuran data sebelumnya, gunakan metode API tingkat rendah. Metode API tingkat rendah untuk unggahan multipart menawarkan fungsionalitas tambahan, untuk informasi selengkapnya, lihat. Menggunakan AWS SDK (API tingkat rendah)

Java

Untuk mengunggah file besar, gunakan `TransferManager` kelas. Operasi API tingkat tinggi ini dapat mengunggah data dari file atau aliran. Anda juga dapat mengatur opsi lanjutan, seperti ukuran bagian yang ingin Anda gunakan untuk unggahan multibagian, atau jumlah thread bersamaan yang ingin Anda gunakan saat mengunggah bagian. Anda juga dapat membuat pengaturan properti objek opsional, kelas penyimpanan, atau daftar kontrol akses (ACL). Anda dapat menggunakan kelas `PutObjectRequest` dan `TransferManagerConfiguration` untuk mengatur opsi lanjutan ini.

Jika memungkinkan, `TransferManager` akan mencoba menggunakan beberapa thread untuk mengunggah beberapa bagian dari satu pengunggahan secara bersamaan. Ketika berhadapan dengan ukuran konten yang besar dan bandwidth yang tinggi, hal ini dapat meningkatkan throughput secara signifikan.

Selain fungsionalitas file-unggahan, kelas `TransferManager` memungkinkan Anda untuk menghentikan unggahan multibagian yang sedang berlangsung. Pengunggahan dianggap sedang berlangsung setelah Anda memulainya dan hingga Anda menyelesaikan atau menghentikannya. `TransferManager` akan menghentikan semua unggahan multibagian yang sedang berlangsung pada sebuah bucket tertentu yang dimulai sebelum tanggal dan waktu yang ditentukan.

> **Note**
>
> Saat Anda menggunakan stream untuk sumber data, kelas `TransferManager` tidak mengunggah secara bersamaan.

Contoh berikut memuat objek menggunakan Java API unggahan multibagian tingkat tinggi (kelas `TransferManager`). Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.transfer.TransferManager;
import com.amazonaws.services.s3.transfer.TransferManagerBuilder;
import com.amazonaws.services.s3.transfer.Upload;

import java.io.File;

public class HighLevelMultipartUpload {

    public static void main(String[] args) throws Exception {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Object key ***";
        String filePath = "*** Path for file to upload ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();
            TransferManager tm = TransferManagerBuilder.standard()
                    .withS3Client(s3Client)
                    .build();

            // TransferManager processes all transfers asynchronously,
            // so this call returns immediately.
```

```
            Upload upload = tm.upload(bucketName, keyName, new File(filePath));
            System.out.println("Object upload started");

            // Optionally, wait for the upload to finish before continuing.
            upload.waitForCompletion();
            System.out.println("Object upload complete");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Untuk mengunggah file ke bucket S3, gunakan kelas `TransferUtility`. Saat mengunggah data dari sebuah file, Anda harus menyediakan nama kunci objeknya. Jika tidak, API akan menggunakan nama file untuk nama kunci. Saat mengunggah data dari stream, Anda harus memberikan nama kunci objek.

Untuk mengatur opsi unggahan tingkat lanjut—seperti ukuran bagian, jumlah thread saat mengunggah beberapa bagian secara bersamaan, metadata, kelas penyimpanan, atau ACL—gunakan kelas `TransferUtilityUploadRequest`.

> **Note**
>
> Saat Anda menggunakan stream untuk sumber data, kelas `TransferUtility` tidak mengunggah secara bersamaan.

Contoh C# berikut mengunggah file ke bucket Amazon S3 dalam beberapa bagian. Contoh ini menunjukkan cara menggunakan berbagai kelebihan muatan `TransferUtility.Upload` untuk mengunggah sebuah file. Setiap panggilan berurutan untuk unggahan menggantikan unggahan sebelumnya. Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Transfer;
using System;
using System.IO;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class UploadFileMPUHighLevelAPITest
    {
        private const string bucketName = "*** provide bucket name ***";
        private const string keyName = "*** provide a name for the uploaded object
 ***";
        private const string filePath = "*** provide the full path name of the file
 to upload ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            UploadFileAsync().Wait();
        }

        private static async Task UploadFileAsync()
        {
            try
            {
                var fileTransferUtility =
                    new TransferUtility(s3Client);

                // Option 1. Upload a file. The file name is used as the object key
 name.
                await fileTransferUtility.UploadAsync(filePath, bucketName);
                Console.WriteLine("Upload 1 completed");

                // Option 2. Specify object key name explicitly.
                await fileTransferUtility.UploadAsync(filePath, bucketName,
 keyName);
                Console.WriteLine("Upload 2 completed");
```

```
                // Option 3. Upload data from a type of System.IO.Stream.
                using (var fileToUpload =
                    new FileStream(filePath, FileMode.Open, FileAccess.Read))
                {
                    await fileTransferUtility.UploadAsync(fileToUpload,
                                                bucketName, keyName);
                }
                Console.WriteLine("Upload 3 completed");

                // Option 4. Specify advanced settings.
                var fileTransferUtilityRequest = new TransferUtilityUploadRequest
                {
                    BucketName = bucketName,
                    FilePath = filePath,
                    StorageClass = S3StorageClass.StandardInfrequentAccess,
                    PartSize = 6291456, // 6 MB.
                    Key = keyName,
                    CannedACL = S3CannedACL.PublicRead
                };
                fileTransferUtilityRequest.Metadata.Add("param1", "Value1");
                fileTransferUtilityRequest.Metadata.Add("param2", "Value2");

                await fileTransferUtility.UploadAsync(fileTransferUtilityRequest);
                Console.WriteLine("Upload 4 completed");
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }

        }
    }
}
```

JavaScript

Example

Unggah file besar.

```
import {
  CreateMultipartUploadCommand,
  UploadPartCommand,
  CompleteMultipartUploadCommand,
  AbortMultipartUploadCommand,
  S3Client,
} from "@aws-sdk/client-s3";

const twentyFiveMB = 25 * 1024 * 1024;

export const createString = (size = twentyFiveMB) => {
  return "x".repeat(size);
};

export const main = async () => {
  const s3Client = new S3Client({});
  const bucketName = "test-bucket";
  const key = "multipart.txt";
  const str = createString();
  const buffer = Buffer.from(str, "utf8");

  let uploadId;

  try {
    const multipartUpload = await s3Client.send(
      new CreateMultipartUploadCommand({
        Bucket: bucketName,
        Key: key,
      }),
    );

    uploadId = multipartUpload.UploadId;

    const uploadPromises = [];
    // Multipart uploads require a minimum size of 5 MB per part.
    const partSize = Math.ceil(buffer.length / 5);

    // Upload each part.
```

```
    for (let i = 0; i < 5; i++) {
      const start = i * partSize;
      const end = start + partSize;
      uploadPromises.push(
        s3Client
          .send(
            new UploadPartCommand({
              Bucket: bucketName,
              Key: key,
              UploadId: uploadId,
              Body: buffer.subarray(start, end),
              PartNumber: i + 1,
            }),
          )
          .then((d) => {
            console.log("Part", i + 1, "uploaded");
            return d;
          }),
      );
    }

    const uploadResults = await Promise.all(uploadPromises);

    return await s3Client.send(
      new CompleteMultipartUploadCommand({
        Bucket: bucketName,
        Key: key,
        UploadId: uploadId,
        MultipartUpload: {
          Parts: uploadResults.map(({ ETag }, i) => ({
            ETag,
            PartNumber: i + 1,
          })),
        },
      }),
    );

    // Verify the output by downloading the file from the Amazon Simple Storage
 Service (Amazon S3) console.
    // Because the output is a 25 MB string, text editors might struggle to open the
 file.
  } catch (err) {
    console.error(err);
```

```
    if (uploadId) {
      const abortCommand = new AbortMultipartUploadCommand({
        Bucket: bucketName,
        Key: key,
        UploadId: uploadId,
      });

      await s3Client.send(abortCommand);
    }
  }
};
```

Example

Unduh file besar.

```
import { GetObjectCommand, S3Client } from "@aws-sdk/client-s3";
import { createWriteStream } from "fs";

const s3Client = new S3Client({});
const oneMB = 1024 * 1024;

export const getObjectRange = ({ bucket, key, start, end }) => {
  const command = new GetObjectCommand({
    Bucket: bucket,
    Key: key,
    Range: `bytes=${start}-${end}`,
  });

  return s3Client.send(command);
};

/**
 * @param {string | undefined} contentRange
 */
export const getRangeAndLength = (contentRange) => {
  const [range, length] = contentRange.split("/");
  const [start, end] = range.split("-");
  return {
    start: parseInt(start),
    end: parseInt(end),
    length: parseInt(length),
  };
};
```

```
export const isComplete = ({ end, length }) => end === length - 1;

// When downloading a large file, you might want to break it down into
// smaller pieces. Amazon S3 accepts a Range header to specify the start
// and end of the byte range to be downloaded.
const downloadInChunks = async ({ bucket, key }) => {
  const writeStream = createWriteStream(
    fileURLToPath(new URL(`./${key}`, import.meta.url)),
  ).on("error", (err) => console.error(err));

  let rangeAndLength = { start: -1, end: -1, length: -1 };

  while (!isComplete(rangeAndLength)) {
    const { end } = rangeAndLength;
    const nextRange = { start: end + 1, end: end + oneMB };

    console.log(`Downloading bytes ${nextRange.start} to ${nextRange.end}`);

    const { ContentRange, Body } = await getObjectRange({
      bucket,
      key,
      ...nextRange,
    });

    writeStream.write(await Body.transformToByteArray());
    rangeAndLength = getRangeAndLength(ContentRange);
  }
};

export const main = async () => {
  await downloadInChunks({
    bucket: "my-cool-bucket",
    key: "my-cool-object.txt",
  });
};
```

Go

Example

Unggah objek besar dengan menggunakan pengelola unggahan untuk memecah data menjadi beberapa bagian dan mengunggahnya secara bersamaan.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3) actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}
```

```
// UploadLargeObject uses an upload manager to upload data to an object in a bucket.
// The upload manager breaks large data into parts and uploads the parts
 concurrently.
func (basics BucketBasics) UploadLargeObject(bucketName string, objectKey string,
 largeObject []byte) error {
 largeBuffer := bytes.NewReader(largeObject)
 var partMiBs int64 = 10
 uploader := manager.NewUploader(basics.S3Client, func(u *manager.Uploader) {
  u.PartSize = partMiBs * 1024 * 1024
 })
 _, err := uploader.Upload(context.TODO(), &s3.PutObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
  Body:   largeBuffer,
 })
 if err != nil {
  log.Printf("Couldn't upload large object to %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }

 return err
}
```

Example

Unduh objek besar dengan menggunakan pengelola unduhan untuk mendapatkan data dalam beberapa bagian dan mengunduhnya secara bersamaan.

```
// DownloadLargeObject uses a download manager to download an object from a bucket.
```

```
// The download manager gets the data in parts and writes them to a buffer until all
 of
// the data has been downloaded.
func (basics BucketBasics) DownloadLargeObject(bucketName string, objectKey string)
 ([]byte, error) {
 var partMiBs int64 = 10
 downloader := manager.NewDownloader(basics.S3Client, func(d *manager.Downloader) {
  d.PartSize = partMiBs * 1024 * 1024
 })
 buffer := manager.NewWriteAtBuffer([]byte{})
 _, err := downloader.Download(context.TODO(), buffer, &s3.GetObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't download large object from %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }
 return buffer.Bytes(), err
}
```

PHP

Topik ini menjelaskan cara menggunakan `Aws\S3\Model\MultipartUpload\UploadBuilder` kelas tingkat tinggi dari AWS SDK for PHP untuk upload file multipart. Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

Contoh PHP berikut mengunggah file ke bucket Amazon S3. Contoh ini menunjukkan cara menetukan parameter untuk objek `MultipartUploader`.

```
 require 'vendor/autoload.php';

use Aws\Exception\MultipartUploadException;
use Aws\S3\MultipartUploader;
use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
```

```
]);

// Prepare the upload parameters.
$uploader = new MultipartUploader($s3, '/path/to/large/file.zip', [
    'bucket' => $bucket,
    'key'    => $keyname
]);

// Perform the upload.
try {
    $result = $uploader->upload();
    echo "Upload complete: {$result['ObjectURL']}" . PHP_EOL;
} catch (MultipartUploadException $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Python

Contoh berikut memuat objek menggunakan API Python unggahan multibagian tingkat tinggi (kelas `TransferManager`).

```
import sys
import threading

import boto3
from boto3.s3.transfer import TransferConfig


MB = 1024 * 1024
s3 = boto3.resource("s3")


class TransferCallback:
    """
    Handle callbacks from the transfer manager.

    The transfer manager periodically calls the __call__ method throughout
    the upload and download process so that it can take action, such as
    displaying progress to the user and collecting data about the transfer.
    """

    def __init__(self, target_size):
```

```
        self._target_size = target_size
        self._total_transferred = 0
        self._lock = threading.Lock()
        self.thread_info = {}

    def __call__(self, bytes_transferred):
        """
        The callback method that is called by the transfer manager.

        Display progress during file transfer and collect per-thread transfer
        data. This method can be called by multiple threads, so shared instance
        data is protected by a thread lock.
        """
        thread = threading.current_thread()
        with self._lock:
            self._total_transferred += bytes_transferred
            if thread.ident not in self.thread_info.keys():
                self.thread_info[thread.ident] = bytes_transferred
            else:
                self.thread_info[thread.ident] += bytes_transferred

            target = self._target_size * MB
            sys.stdout.write(
                f"\r{self._total_transferred} of {target} transferred "
                f"({(self._total_transferred / target) * 100:.2f}%)."
            )
            sys.stdout.flush()


def upload_with_default_configuration(
    local_file_path, bucket_name, object_key, file_size_mb
):
    """
    Upload a file from a local folder to an Amazon S3 bucket, using the default
    configuration.
    """
    transfer_callback = TransferCallback(file_size_mb)
    s3.Bucket(bucket_name).upload_file(
        local_file_path, object_key, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def upload_with_chunksize_and_meta(
```

```
    local_file_path, bucket_name, object_key, file_size_mb, metadata=None
):
    """
    Upload a file from a local folder to an Amazon S3 bucket, setting a
    multipart chunk size and adding metadata to the Amazon S3 object.

    The multipart chunk size controls the size of the chunks of data that are
    sent in the request. A smaller chunk size typically results in the transfer
    manager using more threads for the upload.

    The metadata is a set of key-value pairs that are stored with the object
    in Amazon S3.
    """
    transfer_callback = TransferCallback(file_size_mb)

    config = TransferConfig(multipart_chunksize=1 * MB)
    extra_args = {"Metadata": metadata} if metadata else None
    s3.Bucket(bucket_name).upload_file(
        local_file_path,
        object_key,
        Config=config,
        ExtraArgs=extra_args,
        Callback=transfer_callback,
    )
    return transfer_callback.thread_info


def upload_with_high_threshold(local_file_path, bucket_name, object_key,
 file_size_mb):
    """
    Upload a file from a local folder to an Amazon S3 bucket, setting a
    multipart threshold larger than the size of the file.

    Setting a multipart threshold larger than the size of the file results
    in the transfer manager sending the file as a standard upload instead of
    a multipart upload.
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(multipart_threshold=file_size_mb * 2 * MB)
    s3.Bucket(bucket_name).upload_file(
        local_file_path, object_key, Config=config, Callback=transfer_callback
    )
    return transfer_callback.thread_info
```

```
def upload_with_sse(
    local_file_path, bucket_name, object_key, file_size_mb, sse_key=None
):
    """
    Upload a file from a local folder to an Amazon S3 bucket, adding server-side
    encryption with customer-provided encryption keys to the object.

    When this kind of encryption is specified, Amazon S3 encrypts the object
    at rest and allows downloads only when the expected encryption key is
    provided in the download request.
    """
    transfer_callback = TransferCallback(file_size_mb)
    if sse_key:
        extra_args = {"SSECustomerAlgorithm": "AES256", "SSECustomerKey": sse_key}
    else:
        extra_args = None
    s3.Bucket(bucket_name).upload_file(
        local_file_path, object_key, ExtraArgs=extra_args,
 Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_default_configuration(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, using the
    default configuration.
    """
    transfer_callback = TransferCallback(file_size_mb)
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_single_thread(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, using a
    single thread.
```

```
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(use_threads=False)
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, Config=config, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_high_threshold(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, setting a
    multipart threshold larger than the size of the file.

    Setting a multipart threshold larger than the size of the file results
    in the transfer manager sending the file as a standard download instead
    of a multipart download.
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(multipart_threshold=file_size_mb * 2 * MB)
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, Config=config, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_sse(
    bucket_name, object_key, download_file_path, file_size_mb, sse_key
):
    """
    Download a file from an Amazon S3 bucket to a local folder, adding a
    customer-provided encryption key to the request.

    When this kind of encryption is specified, Amazon S3 encrypts the object
    at rest and allows downloads only when the expected encryption key is
    provided in the download request.
    """
    transfer_callback = TransferCallback(file_size_mb)

    if sse_key:
        extra_args = {"SSECustomerAlgorithm": "AES256", "SSECustomerKey": sse_key}
    else:
```

```
        extra_args = None
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, ExtraArgs=extra_args, Callback=transfer_callback
    )
    return transfer_callback.thread_info
```

Menggunakan AWS SDK (API tingkat rendah)

AWS SDK mengekspos API tingkat rendah yang sangat mirip dengan Amazon S3 REST API untuk unggahan multipart (lihat. Mengunggah dan menyalin objek menggunakan unggahan multibagian Gunakan API tingkat rendah saat Anda perlu menjeda dan melanjutkan unggahan multibagian, memvariasikan ukuran bagian selama pengunggahan, atau tidak mengetahui ukuran data unggahan sebelumnya. Jika Anda tidak memiliki persyaratan ini, gunakan API tingkat tinggi (lihatMenggunakan AWS SDK (API tingkat tinggi)).

Java

Contoh berikut menunjukkan cara menggunakan kelas Java tingkat rendah untuk mengunggah file. Ini melakukan langkah-langkah berikut:

- Memulai unggahan multibagian menggunakan metode `AmazonS3Client.initiateMultipartUpload()`, dan melalui objek `InitiateMultipartUploadRequest`.
- Menyimpan ID unggahan yang dikembalikan oleh metode `AmazonS3Client.initiateMultipartUpload()`. Anda dapat menyediakan ID unggahan untuk setiap operasi unggahan multibagian selanjutnya.
- Mengunggah bagian objek. Untuk setiap bagian, Anda memanggil metode `AmazonS3Client.uploadPart()`. Anda memberikan bagian informasi unggahan menggunakan objek `UploadPartRequest`.
- Untuk setiap bagian, simpan ETag dari respons dari metode `AmazonS3Client.uploadPart()` dalam daftar. Anda dapat menggunakan nilai ETag untuk menyelesaikan unggahan multibagian.
- Memanggil metode `AmazonS3Client.completeMultipartUpload()` untuk menyelesaikan unggahan multibagian.

Example

Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

public class LowLevelMultipartUpload {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ***";
        String filePath = "*** Path to file to upload ***";

        File file = new File(filePath);
        long contentLength = file.length();
        long partSize = 5 * 1024 * 1024; // Set part size to 5 MB.

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Create a list of ETag objects. You retrieve ETags for each object
 part
            // uploaded,
            // then, after each individual part has been uploaded, pass the list of
 ETags to
            // the request to complete the upload.
            List<PartETag> partETags = new ArrayList<PartETag>();
```

```
            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
 InitiateMultipartUploadRequest(bucketName, keyName);
            InitiateMultipartUploadResult initResponse =
 s3Client.initiateMultipartUpload(initRequest);

            // Upload the file parts.
            long filePosition = 0;
            for (int i = 1; filePosition < contentLength; i++) {
                // Because the last part could be less than 5 MB, adjust the part
 size as
                // needed.
                partSize = Math.min(partSize, (contentLength - filePosition));

                // Create the request to upload a part.
                UploadPartRequest uploadRequest = new UploadPartRequest()
                        .withBucketName(bucketName)
                        .withKey(keyName)
                        .withUploadId(initResponse.getUploadId())
                        .withPartNumber(i)
                        .withFileOffset(filePosition)
                        .withFile(file)
                        .withPartSize(partSize);

                // Upload the part and add the response's ETag to our list.
                UploadPartResult uploadResult = s3Client.uploadPart(uploadRequest);
                partETags.add(uploadResult.getPartETag());

                filePosition += partSize;
            }

            // Complete the multipart upload.
            CompleteMultipartUploadRequest compRequest = new
 CompleteMultipartUploadRequest(bucketName, keyName,
                    initResponse.getUploadId(), partETags);
            s3Client.completeMultipartUpload(compRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
```

```
            e.printStackTrace();
        }
    }
}
```

.NET

Contoh C# berikut menunjukkan cara menggunakan API unggahan AWS SDK for .NET multibagian tingkat rendah untuk mengunggah file ke bucket S3. Untuk informasi tentang unggahan multibagian Amazon S3, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.

> **Note**
>
> Saat Anda menggunakan AWS SDK for .NET API untuk mengunggah objek besar, batas waktu mungkin terjadi saat data sedang ditulis ke aliran permintaan. Anda dapat mengatur waktu habis yang eksplisit menggunakan `UploadPartRequest`.

Contoh C# berikut mengunggah file ke bucket S3 menggunakan API unggahan multibagian tingkat rendah. Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.Runtime;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.IO;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class UploadFileMPULowLevelAPITest
    {
        private const string bucketName = "*** provide bucket name ***";
        private const string keyName = "*** provide a name for the uploaded object
 ***";
        private const string filePath = "*** provide the full path name of the file
 to upload ***";
```

```
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            Console.WriteLine("Uploading an object");
            UploadObjectAsync().Wait();
        }

        private static async Task UploadObjectAsync()
        {
            // Create list to store upload part responses.
            List<UploadPartResponse> uploadResponses = new
List<UploadPartResponse>();

            // Setup information required to initiate the multipart upload.
            InitiateMultipartUploadRequest initiateRequest = new
InitiateMultipartUploadRequest
            {
                BucketName = bucketName,
                Key = keyName
            };

            // Initiate the upload.
            InitiateMultipartUploadResponse initResponse =
                await s3Client.InitiateMultipartUploadAsync(initiateRequest);

            // Upload parts.
            long contentLength = new FileInfo(filePath).Length;
            long partSize = 5 * (long)Math.Pow(2, 20); // 5 MB

            try
            {
                Console.WriteLine("Uploading parts");

                long filePosition = 0;
                for (int i = 1; filePosition < contentLength; i++)
                {
                    UploadPartRequest uploadRequest = new UploadPartRequest
                        {
                            BucketName = bucketName,
```

```
                            Key = keyName,
                            UploadId = initResponse.UploadId,
                            PartNumber = i,
                            PartSize = partSize,
                            FilePosition = filePosition,
                            FilePath = filePath
                        };

                    // Track upload progress.
                    uploadRequest.StreamTransferProgress +=
                        new
EventHandler<StreamTransferProgressArgs>(UploadPartProgressEventCallback);

                    // Upload a part and add the response to our list.
                    uploadResponses.Add(await
s3Client.UploadPartAsync(uploadRequest));

                    filePosition += partSize;
                }

                // Setup to complete the upload.
                CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest
                    {
                        BucketName = bucketName,
                        Key = keyName,
                        UploadId = initResponse.UploadId
                     };
                completeRequest.AddPartETags(uploadResponses);

                // Complete the upload.
                CompleteMultipartUploadResponse completeUploadResponse =
                    await s3Client.CompleteMultipartUploadAsync(completeRequest);
            }
            catch (Exception exception)
            {
                Console.WriteLine("An AmazonS3Exception was thrown: { 0}",
exception.Message);

                // Abort the upload.
                AbortMultipartUploadRequest abortMPURequest = new
AbortMultipartUploadRequest
                {
                    BucketName = bucketName,
```

```
                    Key = keyName,
                    UploadId = initResponse.UploadId
                };
               await s3Client.AbortMultipartUploadAsync(abortMPURequest);
            }
        }
        public static void UploadPartProgressEventCallback(object sender,
 StreamTransferProgressArgs e)
        {
            // Process event.
            Console.WriteLine("{0}/{1}", e.TransferredBytes, e.TotalBytes);
        }
    }
}
```

PHP

Topik ini menunjukkan cara menggunakan `uploadPart` metode tingkat rendah dari versi 3 AWS SDK for PHP untuk mengunggah file di beberapa bagian. Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

Contoh PHP berikut mengunggah file ke bucket Amazon S3 menggunakan unggahan multipart PHP API tingkat rendah.

```
 require 'vendor/autoload.php';

use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';
$filename = '*** Path to and Name of the File to Upload ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

$result = $s3->createMultipartUpload([
    'Bucket'       => $bucket,
    'Key'          => $keyname,
    'StorageClass' => 'REDUCED_REDUNDANCY',
    'Metadata'     => [
```

```
        'param1' => 'value 1',
        'param2' => 'value 2',
        'param3' => 'value 3'
    ]
]);
$uploadId = $result['UploadId'];

// Upload the file in parts.
try {
    $file = fopen($filename, 'r');
    $partNumber = 1;
    while (!feof($file)) {
        $result = $s3->uploadPart([
            'Bucket'     => $bucket,
            'Key'        => $keyname,
            'UploadId'   => $uploadId,
            'PartNumber' => $partNumber,
            'Body'       => fread($file, 5 * 1024 * 1024),
        ]);
        $parts['Parts'][$partNumber] = [
            'PartNumber' => $partNumber,
            'ETag' => $result['ETag'],
        ];
        $partNumber++;

        echo "Uploading part $partNumber of $filename." . PHP_EOL;
    }
    fclose($file);
} catch (S3Exception $e) {
    $result = $s3->abortMultipartUpload([
        'Bucket'   => $bucket,
        'Key'      => $keyname,
        'UploadId' => $uploadId
    ]);

    echo "Upload of $filename failed." . PHP_EOL;
}

// Complete the multipart upload.
$result = $s3->completeMultipartUpload([
    'Bucket'   => $bucket,
    'Key'      => $keyname,
    'UploadId' => $uploadId,
    'MultipartUpload'    => $parts,
```

```
]);
$url = $result['Location'];

echo "Uploaded $filename to $url." . PHP_EOL;
```

Menggunakan AWS SDK for Ruby

AWS SDK for Ruby Versi 3 mendukung unggahan multipart Amazon S3 dengan dua cara. Untuk opsi pertama, Anda dapat menggunakan unggahan file terkelola. Untuk informasi lebih lanjut, lihat Mengunggah File ke Amazon S3 dalam AWS Blog Pengembang. Unggahan file terkelola adalah metode yang direkomendasikan untuk mengunggah file ke bucket. Mereka memberikan manfaat berikut:

- Mengelola unggahan multibagian untuk objek yang lebih besar dari 15 MB.
- Membuka file dengan benar dalam mode biner untuk menghindari masalah pengodean.
- Menggunakan beberapa thread untuk mengunggah bagian-bagian objek besar secara paralel.

Alternatifnya, Anda dapat menggunakan operasi klien unggahan multibagian berikut secara langsung:

- create_multipart_upload–Memulai sebuah unggahan multibagian dan menampilkan sebuah ID unggahan.
- upload_part–Mengunggah satu bagian dalam sebuah unggahan multibagian.
- upload_part_copy–Mengunggah satu bagian dengan menyalin data dari objek yang ada sebagai sumber data.
- complete_multipart_upload–Menyelesaikan sebuah unggahan multibagian dengan merangkai bagian-bagian yang diunggah sebelumnya.
- abort_multipart_upload–Menghentikan unggahan multibagian.

Penggunaan API REST

Bagian berikut dalam Referensi API Amazon Simple Storage Service menjelaskan tentang API REST untuk unggahan multibagian.

- Mulai Unggahan Multibagian
- Unggah Bagian

- Selesaikan Unggahan Multibagian
- Hentikan Unggahan Multibagian
- Daftarkan Bagian
- Daftarkan Unggahan Multibagian

Menggunakan AWS CLI

Bagian berikut dalam AWS Command Line Interface (AWS CLI) menjelaskan operasi untuk upload multipart.

- Mulai Unggahan Multibagian
- Unggah Bagian
- Unggah Bagian (Salinan)
- Selesaikan Unggahan Multibagian
- Batalkan Unggahan Multibagian
- Daftarkan Bagian
- Daftarkan Unggahan Multibagian

Anda juga dapat menggunakan API REST untuk membuat permintaan REST Anda sendiri, atau Anda dapat menggunakan salah satu SDK AWS . Untuk informasi selengkapnya tentang API REST, lihat Penggunaan API REST. Untuk informasi selengkapnya tentang SDK, lihat Pengunggahan objek menggunakan unggahan multibagian.

Mengunggah direktori menggunakan kelas .NET tingkat tinggi TransferUtility

Anda dapat menggunakan kelas `TransferUtility` untuk mengunggah sebuah direktori secara keseluruhan. Secara default, API hanya mengunggah file di dalam root direktori yang ditentukan. Namun, Anda dapat menentukan pengunggahan file secara berulang-ulang dalam semua subdirektori.

Untuk memilih file di direktori tertentu berdasarkan kriteria pemfilteran, tentukan ekspresi pemfilterannya. Misalnya, untuk mengunggah file .pdf saja dari sebuah direktori, tentukan ekspresi filter `"*.pdf"`.

Saat mengunggah file dari sebuah direktori, Anda tidak perlu menentukan nama kunci untuk objek yang dihasilkan. Amazon S3 membangun nama kunci menggunakan jalur file asli. Misalnya, asumsikan bahwa Anda memiliki sebuah direktori yang disebut sebagai `c:\myfolder` dengan struktur berikut:

Example

```
C:\myfolder
      \a.txt
      \b.pdf
      \media\
             An.mp3
```

Saat Anda mengunggah direktori ini, Amazon S3 menggunakan nama kunci berikut:

Example

```
a.txt
b.pdf
media/An.mp3
```

Example

Contoh C# berikut mengunggah sebuah direktori ke bucket Amazon S3. Contoh ini menunjukkan cara menggunakan berbagai kelebihan muatan `TransferUtility.UploadDirectory` untuk mengunggah direktori. Setiap panggilan berurutan untuk unggahan menggantikan unggahan sebelumnya. Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Transfer;
using System;
using System.IO;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class UploadDirMPUHighLevelAPITest
    {
        private const string existingBucketName = "*** bucket name ***";
        private const string directoryPath = @"*** directory path ***";
```

```
        // The example uploads only .txt files.
        private const string wildCard = "*.txt";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            UploadDirAsync().Wait();
        }

        private static async Task UploadDirAsync()
        {
            try
            {
                var directoryTransferUtility =
                    new TransferUtility(s3Client);

                // 1. Upload a directory.
                await directoryTransferUtility.UploadDirectoryAsync(directoryPath,
                    existingBucketName);
                Console.WriteLine("Upload statement 1 completed");

                // 2. Upload only the .txt files from a directory
                //    and search recursively.
                await directoryTransferUtility.UploadDirectoryAsync(
                                                directoryPath,
                                                existingBucketName,
                                                wildCard,
                                                SearchOption.AllDirectories);
                Console.WriteLine("Upload statement 2 completed");

                // 3. The same as Step 2 and some optional configuration.
                //    Search recursively for .txt files to upload.
                var request = new TransferUtilityUploadDirectoryRequest
                {
                    BucketName = existingBucketName,
                    Directory = directoryPath,
                    SearchOption = SearchOption.AllDirectories,
                    SearchPattern = wildCard
                };

                await directoryTransferUtility.UploadDirectoryAsync(request);
                Console.WriteLine("Upload statement 3 completed");
```

```
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine(
                        "Error encountered ***. Message:'{0}' when writing an object",
 e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine(
                    "Unknown encountered on server. Message:'{0}' when writing an
 object", e.Message);
            }
        }
    }
}
```

Mendaftarkan unggahan multibagian

Anda dapat menggunakan AWS SDK (API tingkat rendah) untuk mengambil daftar unggahan multipart yang sedang berlangsung di Amazon S3.

Mencantumkan unggahan multibagian menggunakan AWS SDK (API tingkat rendah)

Java

Tugas berikut memandu Anda dalam menggunakan kelas Java tingkat rendah untuk mencantumkan semua unggahan multibagian yang sedang berlangsung di dalam sebuah bucket.

Proses pendaftaran unggahan multibagian API tingkat rendah

1	Membuat sebuah instans dari kelas `ListMultipartUploadsRequest` dan memberikan nama bucket.
2	Jalankan metode `AmazonS3Client.listMultipartUploads`. Metode tersebut akan menampilkan sebuah instans dari kelas `MultipartUploadListing` yang menyediakan informasi tentang unggahan multibagian yang sedang berlangsung.

Contoh kode Java berikut menunjukkan tugas sebelumnya.

Example

```
ListMultipartUploadsRequest allMultpartUploadsRequest =
     new ListMultipartUploadsRequest(existingBucketName);
MultipartUploadListing multipartUploadListing =
     s3Client.listMultipartUploads(allMultpartUploadsRequest);
```

.NET

Untuk mendaftarkan semua unggahan multibagian yang sedang berlangsung pada bucket tertentu, gunakan kelas AWS SDK for .NET API unggahan multibagian tingkat rendah `ListMultipartUploadsRequest`. Metode `AmazonS3Client.ListMultipartUploads` akan mengembalikan instans dari kelas `ListMultipartUploadsResponse` yang memberikan informasi tentang unggahan multibagian yang sedang berlangsung.

Unggahan multibagian yang sedang berlangsung adalah unggahan multibagian yang telah dimulai menggunakan permintaan unggahan multibagian, namun belum diselesaikan atau dihentikan. Untuk informasi selengkapnya tentang unggahan multipart Amazon S3, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.

Contoh C# berikut menunjukkan cara menggunakan daftar semua AWS SDK for .NET unggahan multipart yang sedang berlangsung di bucket. Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
ListMultipartUploadsRequest request = new ListMultipartUploadsRequest
{
  BucketName = bucketName // Bucket receiving the uploads.
};

ListMultipartUploadsResponse response = await
 AmazonS3Client.ListMultipartUploadsAsync(request);
```

PHP

Topik ini menunjukkan cara menggunakan class API tingkat rendah dari versi 3 AWS SDK for PHP untuk mencantumkan semua unggahan multibagian yang sedang berlangsung di bucket. Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

Contoh PHP berikut mendemonstrasikan pendaftaran semua unggahan multibagian yang sedang berlangsung pada sebuah bucket.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Retrieve a list of the current multipart uploads.
$result = $s3->listMultipartUploads([
    'Bucket' => $bucket
]);

// Write the list of uploads to the page.
print_r($result->toArray());
```

Mendaftarkan API REST untuk unggahan multibagian

Bagian dalam Referensi API Amazon Simple Storage Service berikut ini menjelaskan tentang API REST untuk unggahan multibagian:

- ListParts-daftar bagian yang diunggah untuk unggahan multibagian tertentu.
- ListMultipartUploads-daftar unggahan multipart yang sedang berlangsung.

Membuat daftar unggahan multipart menggunakan AWS CLI

Bagian berikut dalam AWS Command Line Interface menjelaskan operasi untuk daftar unggahan multipart.

- list-parts-mendaftarkan bagian yang terunggah untuk unggahan multibagian tertentu.
- list-multipart-uploads-daftar unggahan multipart yang sedang berlangsung.

Melacak unggahan multibagian

API unggahan multibagian tingkat tinggi menyediakan antarmuka dengar, `ProgressListener`, untuk melacak progres saat mengunggah sebuah objek ke Amazon S3. Peristiwa progres secara berkala memberi tahu pendengar bahwa byte telah ditransfer.

Java

Example

```
TransferManager tm = new TransferManager(new ProfileCredentialsProvider());

PutObjectRequest request = new PutObjectRequest(
    existingBucketName, keyName, new File(filePath));

// Subscribe to the event and provide event handler.
request.setProgressListener(new ProgressListener() {
   public void progressChanged(ProgressEvent event) {
    System.out.println("Transferred bytes: " +
      event.getBytesTransfered());
             }
});
```

Example

Kode Java berikut mengunggah file dan menggunakan `ProgressListener` untuk melacak kemajuan unggahan. Untuk petunjuk tentang cara membuat dan menguji sampel kerja, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import java.io.File;

import com.amazonaws.AmazonClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.event.ProgressEvent;
import com.amazonaws.event.ProgressListener;
import com.amazonaws.services.s3.model.PutObjectRequest;
import com.amazonaws.services.s3.transfer.TransferManager;
import com.amazonaws.services.s3.transfer.Upload;

public class TrackMPUProgressUsingHighLevelAPI {

    public static void main(String[] args) throws Exception {
```

```
        String existingBucketName = "*** Provide bucket name ***";
        String keyName            = "*** Provide object key ***";
        String filePath           = "*** file to upload ***";

        TransferManager tm = new TransferManager(new ProfileCredentialsProvider());


        // For more advanced uploads, you can create a request object
        // and supply additional request parameters (ex: progress listeners,
        // canned ACLs, etc.)
        PutObjectRequest request = new PutObjectRequest(
          existingBucketName, keyName, new File(filePath));

        // You can ask the upload for its progress, or you can
        // add a ProgressListener to your request to receive notifications
        // when bytes are transferred.
        request.setGeneralProgressListener(new ProgressListener() {
   @Override
   public void progressChanged(ProgressEvent progressEvent) {
    System.out.println("Transferred bytes: " +
      progressEvent.getBytesTransferred());
   }
  });

        // TransferManager processes all transfers asynchronously,
        // so this call will return immediately.
        Upload upload = tm.upload(request);

        try {
         // You can block and wait for the upload to finish
         upload.waitForCompletion();
        } catch (AmazonClientException amazonClientException) {
         System.out.println("Unable to upload file, upload aborted.");
         amazonClientException.printStackTrace();
        }
    }
}
```

.NET

Contoh C# berikut ini menampilkan mengunggah sebuah file ke sebuah bucket S3 menggunakan kelas `TransferUtility`, dan melacak progres unggahan. Untuk informasi tentang menyiapkan

dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Transfer;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class TrackMPUUsingHighLevelAPITest
    {
        private const string bucketName = "*** provide the bucket name ***";
        private const string keyName = "*** provide the name for the uploaded object
 ***";
        private const string filePath = " *** provide the full path name of the file
 to upload **";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;


        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            TrackMPUAsync().Wait();
        }

        private static async Task TrackMPUAsync()
        {
            try
            {
                var fileTransferUtility = new TransferUtility(s3Client);

                // Use TransferUtilityUploadRequest to configure options.
                // In this example we subscribe to an event.
                var uploadRequest =
                    new TransferUtilityUploadRequest
                    {
                        BucketName = bucketName,
                        FilePath = filePath,
```

```
                        Key = keyName
                    };

                uploadRequest.UploadProgressEvent +=
                    new EventHandler<UploadProgressArgs>
                        (uploadRequest_UploadPartProgressEvent);

                await fileTransferUtility.UploadAsync(uploadRequest);
                Console.WriteLine("Upload completed");
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }

        static void uploadRequest_UploadPartProgressEvent(object sender,
 UploadProgressArgs e)
        {
            // Process event.
            Console.WriteLine("{0}/{1}", e.TransferredBytes, e.TotalBytes);
        }
    }
}
```

Membatalkan unggahan multibagian

Setelah memulai unggahan multibagian, Anda memulai pengunggahan bagian. Amazon S3 menyimpan bagian-bagian ini, namun membuat objek dari bagian-bagian tersebut hanya setelah Anda mengunggah semuanya dan mengirimkan permintaan `successful` untuk menyelesaikan unggahan multibagian (Anda harus memverifikasi bahwa permintaan Anda untuk menyelesaikan unggahan multibagian berhasil). Setelah menerima keseluruhan permintaan unggahan multibagian, Amazon S3 akan menyusun bagian-bagiannya, dan membuat sebuah objek. Jika Anda tidak berhasil mengirimkan permintaan unggahan multibagian lengkap, Amazon S3 tidak merakit bagian-bagiannya, dan tidak membuat objek apa pun.

Anda akan ditagih untuk semua penyimpanan yang terkait dengan bagian yang diunggah. Untuk informasi selengkapnya, lihat Unggahan multibagian dan harga. Sehingga, Anda harus menyelesaikan unggahan multibagian untuk memiliki objek dibuat atau menghentikan unggahan multibagian untuk membuang bagian yang telah diunggah.

Anda dapat menghentikan unggahan multipart yang sedang berlangsung di Amazon S3 menggunakan AWS Command Line Interface AWS CLI(), REST API, atau SDK. AWS Anda juga dapat menghentikan unggahan multibagian yang tidak lengkap menggunakan konfigurasi siklus hidup bucket.

Menggunakan AWS SDK (API tingkat tinggi)

Java

Kelas `TransferManager` menyediakan metode `abortMultipartUploads` untuk menghentikan unggahan multibagian yang sedang berlangsung. Pengunggahan dianggap sedang berlangsung setelah Anda memulainya, dan hingga Anda menyelesaikan atau menghentikannya. Anda memberikan nilai `Date`, dan API ini menghentikan semua unggahan multibagian pada bucket yang dimulai sebelum `Date` yang ditentukan dan masih dalam proses.

Tugas berikut memandu Anda dalam menggunakan kelas Java tingkat tinggi untuk menghentikan pengunggahan multibagian.

Proses penghentian unggahan multibagian API tingkat tinggi

1	Buat instans dari kelas `TransferManager` .
2	Jalankan metode `TransferManager.abortMultipartUploads` dengan meneruskan nama bucket dan nilai `Date`.

Contoh kode Java berikut menghentikan semua unggahan multibagian yang sedang berlangsung yang dimulai pada bucket tertentu lebih dari satu minggu yang lalu. Untuk petunjuk tentang cara membuat dan menguji sampel kerja, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import java.util.Date;

import com.amazonaws.AmazonClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3.transfer.TransferManager;
```

```
public class AbortMPUUsingHighLevelAPI {

    public static void main(String[] args) throws Exception {
        String existingBucketName = "*** Provide existing bucket name ***";

        TransferManager tm = new TransferManager(new ProfileCredentialsProvider());


        int sevenDays = 1000 * 60 * 60 * 24 * 7;
  Date oneWeekAgo = new Date(System.currentTimeMillis() - sevenDays);

        try {
         tm.abortMultipartUploads(existingBucketName, oneWeekAgo);
        } catch (AmazonClientException amazonClientException) {
         System.out.println("Unable to upload file, upload was aborted.");
         amazonClientException.printStackTrace();
        }
    }
}
```

> **Note**
>
> Anda juga dapat menghentikan unggahan multibagian tertentu. Untuk informasi selengkapnya, lihat Menggunakan AWS SDK (API tingkat rendah).

.NET

Contoh C# berikut menghentikan semua unggahan multibagian yang sedang berlangsung yang dimulai pada bucket tertentu lebih dari seminggu yang lalu. Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Transfer;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class AbortMPUUsingHighLevelAPITest
```

```
    {
        private const string bucketName = "*** provide bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            AbortMPUAsync().Wait();
        }

        private static async Task AbortMPUAsync()
        {
            try
            {
                var transferUtility = new TransferUtility(s3Client);

                // Abort all in-progress uploads initiated before the specified
 date.
                await transferUtility.AbortMultipartUploadsAsync(
                    bucketName, DateTime.Now.AddDays(-7));
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

> **Note**
>
> Anda juga dapat menghentikan unggahan multibagian tertentu. Untuk informasi selengkapnya, lihat Menggunakan AWS SDK (API tingkat rendah).

Menggunakan AWS SDK (API tingkat rendah)

Anda dapat menghentikan unggahan multibagian yang sedang berlangsung dengan memanggil metode `AmazonS3.abortMultipartUpload`. Metode ini menghapus semua bagian yang diunggah pada Amazon S3, dan mengosongkan sumber daya. Anda harus menyediakan ID unggahan, nama bucket, dan nama kunci. Contoh kode Java berikut menunjukkan cara menghentikan sebuah unggahan multibagian yang sedang berlangsung.

Untuk menghentikan unggahan multibagian, Anda harus memberikan ID unggahan, dan bucket serta nama kunci yang digunakan dalam unggahan tersebut. Setelah Anda menghentikan unggahan multibagian, Anda tidak dapat menggunakan ID unggahan untuk mengunggah bagian tambahan. Untuk informasi selengkapnya tentang unggahan multibagian Amazon S3, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.

Java

Contoh kode Java berikut menghentikan unggahan multibagian yang sedang berlangsung.

Example

```
InitiateMultipartUploadRequest initRequest =
    new InitiateMultipartUploadRequest(existingBucketName, keyName);
InitiateMultipartUploadResult initResponse =
               s3Client.initiateMultipartUpload(initRequest);

AmazonS3 s3Client = new AmazonS3Client(new ProfileCredentialsProvider());
s3Client.abortMultipartUpload(new AbortMultipartUploadRequest(
            existingBucketName, keyName, initResponse.getUploadId()));
```

> **Note**
>
> Alih-alih melakukan unggahan multibagian tertentu, Anda dapat menghentikan semua unggahan multibagian yang dimulai sebelum waktu tertentu yang masih berlangsung. Operasi pembersihan ini berguna untuk menghentikan unggahan multibagian lama yang Anda mulai tetapi tidak diselesaikan atau dihentikan. Untuk informasi selengkapnya, lihat Menggunakan AWS SDK (API tingkat tinggi).

.NET

Contoh C# berikut menunjukkan cara untuk menghentikan unggahan multibagian. Untuk keseluruhan sampel C# yang mencakup kode berikut, lihat Menggunakan AWS SDK (API tingkat rendah).

```
AbortMultipartUploadRequest abortMPURequest = new AbortMultipartUploadRequest
{
    BucketName = existingBucketName,
    Key = keyName,
    UploadId = initResponse.UploadId
};
await AmazonS3Client.AbortMultipartUploadAsync(abortMPURequest);
```

Anda juga dapat membatalkan semua unggahan multibagian yang sedang berlangsung, yang dimulai sebelum waktu tertentu. Operasi pembersihan ini berguna untuk membatalkan unggahan multibagian yang tidak selesai atau dibatalkan. Untuk informasi selengkapnya, lihat Menggunakan AWS SDK (API tingkat tinggi).

PHP

Contoh ini menunjukkan cara menggunakan kelas dari versi 3 AWS SDK for PHP untuk membatalkan unggahan multipart yang sedang berlangsung. Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2. Contoh metode `abortMultipartUpload()`.

Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';
$uploadId = '*** Upload ID of upload to Abort ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);
```

```
// Abort the multipart upload.
$s3->abortMultipartUpload([
    'Bucket'   => $bucket,
    'Key'      => $keyname,
    'UploadId' => $uploadId,
]);
```

Penggunaan API REST

Untuk informasi selengkapnya tentang penggunaan REST API untuk menghentikan unggahan multibagian, lihat AbortMultipartUploaddi Referensi API Amazon Simple Storage Service.

Menggunakan AWS CLI

Untuk informasi selengkapnya tentang penggunaan AWS CLI untuk menghentikan unggahan multibagian, lihat abort-multipart-uploaddi Referensi AWS CLI Perintah.

Menyalin objek menggunakan unggahan multibagian

Contoh dalam bagian ini menunjukkan cara menyalin objek yang lebih besar dari 5 GB menggunakan API unggahan multibagian. Anda dapat menyalin objek kurang dari 5 GB dalam satu operasi. Untuk informasi selengkapnya, lihat Menyalin, memindahkan, dan mengganti nama objek.

Menggunakan AWS SDK

Untuk menyalin sebuah objek menggunakan API tingkat rendah, lakukan hal berikut ini:

- Mulai unggahan multibagian dengan memanggil metode `AmazonS3Client.initiateMultipartUpload()`.
- Simpan ID unggahan dari objek respons yang dikembalikan oleh metode `AmazonS3Client.initiateMultipartUpload()`. Anda memberikan ID unggahan ini untuk setiap operasi unggahan bagian.
- Salin semua bagiannya. Untuk setiap bagian yang harus Anda salin, buat sebuah instans baru dari kelas `CopyPartRequest`. Berikan informasi bagian, termasuk nama bucket sumber dan tujuan, kunci objek sumber dan tujuan, ID unggahan, lokasi byte pertama dan terakhir bagian tersebut, dan nomor bagian.
- Simpan respons dari panggilan metode `AmazonS3Client.copyPart()`. Setiap respons mencakup nilai `ETag` dan nomor bagian untuk bagian yang diunggah. Anda memerlukan informasi ini untuk menyelesaikan unggahan multibagian.

- Perintahkan metode `AmazonS3Client.completeMultipartUpload()` untuk menyelesaikan operasi penyalinan.

Java

Example

Contoh berikut ini menunjukkan cara menggunakan Java API tingkat rendah Amazon S3 untuk melakukan penyalinan multibagian. Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

public class LowLevelMultipartCopy {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String sourceBucketName = "*** Source bucket name ***";
        String sourceObjectKey = "*** Source object key ***";
        String destBucketName = "*** Target bucket name ***";
        String destObjectKey = "*** Target object key ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
 InitiateMultipartUploadRequest(destBucketName,
```

```
                    destObjectKey);
            InitiateMultipartUploadResult initResult =
s3Client.initiateMultipartUpload(initRequest);

            // Get the object size to track the end of the copy operation.
            GetObjectMetadataRequest metadataRequest = new
GetObjectMetadataRequest(sourceBucketName, sourceObjectKey);
            ObjectMetadata metadataResult =
s3Client.getObjectMetadata(metadataRequest);
            long objectSize = metadataResult.getContentLength();

            // Copy the object using 5 MB parts.
            long partSize = 5 * 1024 * 1024;
            long bytePosition = 0;
            int partNum = 1;
            List<CopyPartResult> copyResponses = new ArrayList<CopyPartResult>();
            while (bytePosition < objectSize) {
                // The last part might be smaller than partSize, so check to make
sure
                // that lastByte isn't beyond the end of the object.
                long lastByte = Math.min(bytePosition + partSize - 1, objectSize -
1);

                // Copy this part.
                CopyPartRequest copyRequest = new CopyPartRequest()
                        .withSourceBucketName(sourceBucketName)
                        .withSourceKey(sourceObjectKey)
                        .withDestinationBucketName(destBucketName)
                        .withDestinationKey(destObjectKey)
                        .withUploadId(initResult.getUploadId())
                        .withFirstByte(bytePosition)
                        .withLastByte(lastByte)
                        .withPartNumber(partNum++);
                copyResponses.add(s3Client.copyPart(copyRequest));
                bytePosition += partSize;
            }

            // Complete the upload request to concatenate all uploaded parts and
make the
            // copied object available.
            CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest(
                    destBucketName,
                    destObjectKey,
```

```
                    initResult.getUploadId(),
                    getETags(copyResponses));
            s3Client.completeMultipartUpload(completeRequest);
            System.out.println("Multipart copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    // This is a helper function to construct a list of ETags.
    private static List<PartETag> getETags(List<CopyPartResult> responses) {
        List<PartETag> etags = new ArrayList<PartETag>();
        for (CopyPartResult response : responses) {
            etags.add(new PartETag(response.getPartNumber(), response.getETag()));
        }
        return etags;
    }
}
```

.NET

Contoh C # berikut menunjukkan cara menggunakan AWS SDK for .NET untuk menyalin objek Amazon S3 yang lebih besar dari 5 GB dari satu lokasi sumber ke lokasi lain, seperti dari satu bucket ke bucket lainnya. Untuk menyalin objek yang lebih kecil dari 5 GB, gunakan prosedur penyalinan operasi tunggal yang dijelaskan dalam Menggunakan AWS SDK. Untuk informasi selengkapnya tentang unggahan multibagian Amazon S3, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.

Contoh ini menunjukkan cara menyalin objek Amazon S3 yang lebih besar dari 5 GB dari satu bucket S3 ke bucket lainnya menggunakan API unggahan AWS SDK for .NET multipart.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
```

```
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CopyObjectUsingMPUapiTest
    {
        private const string sourceBucket = "*** provide the name of the bucket with
 source object ***";
        private const string targetBucket = "*** provide the name of the bucket to
 copy the object to ***";
        private const string sourceObjectKey = "*** provide the name of object to
 copy ***";
        private const string targetObjectKey = "*** provide the name of the object
 copy ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            Console.WriteLine("Copying an object");
            MPUCopyObjectAsync().Wait();
        }
        private static async Task MPUCopyObjectAsync()
        {
            // Create a list to store the upload part responses.
            List<UploadPartResponse> uploadResponses = new
 List<UploadPartResponse>();
            List<CopyPartResponse> copyResponses = new List<CopyPartResponse>();

            // Setup information required to initiate the multipart upload.
            InitiateMultipartUploadRequest initiateRequest =
                new InitiateMultipartUploadRequest
                {
                    BucketName = targetBucket,
                    Key = targetObjectKey
                };

            // Initiate the upload.
            InitiateMultipartUploadResponse initResponse =
                await s3Client.InitiateMultipartUploadAsync(initiateRequest);
```

```
            // Save the upload ID.
            String uploadId = initResponse.UploadId;

            try
            {
                // Get the size of the object.
                GetObjectMetadataRequest metadataRequest = new
GetObjectMetadataRequest
                {
                    BucketName = sourceBucket,
                    Key = sourceObjectKey
                };

                GetObjectMetadataResponse metadataResponse =
                    await s3Client.GetObjectMetadataAsync(metadataRequest);
                long objectSize = metadataResponse.ContentLength; // Length in
bytes.

                // Copy the parts.
                long partSize = 5 * (long)Math.Pow(2, 20); // Part size is 5 MB.

                long bytePosition = 0;
                for (int i = 1; bytePosition < objectSize; i++)
                {
                    CopyPartRequest copyRequest = new CopyPartRequest
                    {
                        DestinationBucket = targetBucket,
                        DestinationKey = targetObjectKey,
                        SourceBucket = sourceBucket,
                        SourceKey = sourceObjectKey,
                        UploadId = uploadId,
                        FirstByte = bytePosition,
                        LastByte = bytePosition + partSize - 1 >= objectSize ?
objectSize - 1 : bytePosition + partSize - 1,
                        PartNumber = i
                    };

                    copyResponses.Add(await s3Client.CopyPartAsync(copyRequest));

                    bytePosition += partSize;
                }

                // Set up to complete the copy.
                CompleteMultipartUploadRequest completeRequest =
```

```
                new CompleteMultipartUploadRequest
                {
                    BucketName = targetBucket,
                    Key = targetObjectKey,
                    UploadId = initResponse.UploadId
                };
                completeRequest.AddPartETags(copyResponses);

                // Complete the copy.
                CompleteMultipartUploadResponse completeUploadResponse =
                    await s3Client.CompleteMultipartUploadAsync(completeRequest);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

Penggunaan API REST

Bagian berikut dalam Referensi API Amazon Simple Storage Service menjelaskan tentang API REST untuk unggahan multibagian. Untuk menyalin objek yang sudah ada, Anda dapat menggunakan API Unggah Bagian (Salinan), dan menentukan objek sumber dengan menambahkan header permintaan `x-amz-copy-source` dalam permintaan Anda.

- Mulai Unggahan Multibagian
- Unggah Bagian
- Unggah Bagian (Salinan)
- Selesaikan Unggahan Multibagian
- Batalkan Unggahan Multibagian
- Daftarkan Bagian

- Daftarkan Unggahan Multibagian

Anda dapat menggunakan API ini untuk membuat permintaan REST Anda sendiri, atau Anda dapat menggunakan SDK yang kami sediakan. Untuk informasi selengkapnya tentang menggunakan Unggahan Multipart dengan AWS CLI, lihatMenggunakan AWS CLI. Untuk informasi selengkapnya tentang SDK, lihat AWS Dukungan SDK untuk unggahan multipart.

Batas unggahan multibagian Amazon S3

Tabel berikut ini menyediakan spesifikasi inti unggahan multibagian. Untuk informasi selengkapnya, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.

Item	Spesifikasi
Ukuran objek maksimum	5 TiB
Jumlah maksimum bagian per unggahan	10.000
Nomor bagian	1 hingga 10.000 (inklusif)
Ukuran bagian	5 MiB hingga 5 GiB. Tidak ada batas ukuran minimum pada bagian terakhir dari unggahan multibagian Anda.
Jumlah maksimum bagian yang ditampilkan untuk permintaan daftar bagian	1000
Jumlah maksimum unggahan multibagian yang dikembalikan dalam permintaan unggahan multibagian daftar	1000

Menyalin, memindahkan, dan mengganti nama objek

`CopyObject`Operasi membuat salinan objek yang sudah disimpan di Amazon S3.

Anda dapat membuat salinan objek hingga 5 GB dalam satu operasi atom. Namun, untuk menyalin objek yang lebih besar dari 5 GB, Anda harus menggunakan unggahan multipart. Untuk informasi selengkapnya, lihat the section called “Menyalin objek”.

Dengan menggunakan operasi `CopyObject`, Anda dapat:

- Buat salinan objek tambahan.
- Ganti nama objek dengan menyalinnya dan menghapus yang asli.
- Salin atau pindahkan objek dari satu ember ke ember lainnya, termasuk melintasi Wilayah AWS (misalnya, dari `us-west-1` keeu-`west-2`). Saat Anda memindahkan objek, Amazon S3 menyalin objek ke tujuan yang ditentukan dan kemudian menghapus objek sumber.

  > **Note**
  >
  > Menyalin atau memindahkan objek di seluruh Wilayah AWS menimbulkan biaya bandwidth. Untuk informasi selengkapnya, lihat harga Amazon S3.

- Ubah metadata objek. Setiap objek Amazon S3 memiliki metadata. Metadata ini adalah satu set pasangan nama-nilai. Anda dapat mengatur metadata objek pada saat Anda mengunggah objek. Setelah Anda mengunggah objek, Anda tidak dapat mengubah metadata objek. Satu-satunya cara untuk memodifikasi metadata objek adalah membuat salinan objek, dan mengatur metadatanya. Untuk melakukannya, dalam operasi penyalinan, atur objek yang sama dengan sumber dan target.

  Beberapa metadata objek adalah metadata sistem dan lainnya ditentukan pengguna. Anda dapat mengontrol beberapa metadata sistem. Misalnya, Anda dapat mengontrol kelas penyimpanan dan jenis enkripsi sisi server yang akan digunakan untuk objek. Saat Anda menyalin sebuah objek, metadata sistem yang dikontrol pengguna dan metadata yang ditentukan pengguna juga disalin. Amazon S3 mengatur ulang metadata yang dikontrol sistem. Misalnya, saat Anda menyalin suatu objek, Amazon S3 mengatur ulang tanggal penciptaan objek yang disalin. Anda tidak perlu menyetel salah satu nilai metadata yang dikendalikan sistem ini dalam permintaan salinan Anda.

  Saat menyalin sebuah objek, Anda mungkin memutuskan untuk memperbarui beberapa nilai metadata. Misalnya, jika objek sumber Anda dikonfigurasi untuk menggunakan penyimpanan Standar S3, Anda dapat memilih untuk menggunakan S3 Intelligent-Tiering untuk salinan objek tersebut. Anda juga dapat memutuskan untuk mengubah beberapa nilai metadata yang ditentukan pengguna yang terdapat pada objek sumber tersebut. Jika Anda memilih untuk memperbarui salah satu metadata objek yang dapat dikonfigurasi pengguna (sistem atau yang ditentukan pengguna) selama penyalinan, maka Anda harus secara eksplisit menentukan semua metadata yang dapat

dikonfigurasi pengguna yang ada pada objek sumber dalam permintaan Anda, bahkan jika Anda mengubahnya hanya satu dari nilai metadata.

Untuk informasi selengkapnya tentang metadata objek, lihat Bekerja dengan metadata objek.

Menyalin objek yang diarsipkan dan dipulihkan

Jika objek sumber diarsipkan di S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, Anda harus memulihkan salinan sementara terlebih dahulu sebelum dapat menyalin objek ke bucket lain. Untuk informasi tentang pengarsipan objek, lihat Transisi ke kelas penyimpanan S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive (pengarsipan objek).

Operasi Salin di konsol Amazon S3 tidak didukung untuk objek yang dipulihkan di kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive. Untuk menyalin objek yang dipulihkan ini, gunakan AWS Command Line Interface (AWS CLI), AWS SDK, atau Amazon S3 REST API.

Menyalin objek terenkripsi

Amazon S3 secara otomatis mengenkripsi semua objek baru yang disalin ke bucket S3. Jika Anda tidak menentukan informasi enkripsi dalam permintaan penyalinan, pengaturan enkripsi objek target diatur ke konfigurasi enkripsi default bucket tujuan. Secara default, semua bucket memiliki konfigurasi enkripsi tingkat dasar yang menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3). Jika bucket tujuan memiliki konfigurasi enkripsi default yang menggunakan enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS), atau kunci enkripsi yang disediakan pelanggan (SSE-C), Amazon S3 menggunakan kunci KMS yang sesuai, atau kunci yang disediakan pelanggan untuk mengenkripsi salinan objek target.

Saat menyalin objek, jika Anda ingin menggunakan jenis pengaturan enkripsi yang berbeda untuk objek target, Anda dapat meminta Amazon S3 mengenkripsi objek target dengan kunci KMS, kunci terkelola Amazon S3, atau kunci yang disediakan pelanggan. Jika pengaturan enkripsi dalam permintaan Anda berbeda dari konfigurasi enkripsi default bucket tujuan, pengaturan enkripsi dalam permintaan Anda akan lebih diutamakan. Jika objek sumber untuk salinan dienkripsi dengan SSE-C, Anda harus memberikan informasi enkripsi yang diperlukan dalam permintaan Anda sehingga Amazon S3 dapat mendekripsi objek untuk disalin. Untuk informasi selengkapnya, lihat Melindungi data dengan enkripsi.

Menggunakan checksum saat menyalin objek

Saat menyalin objek, Anda dapat memilih untuk menggunakan algoritma checksum yang berbeda untuk objek tersebut. Apakah Anda memilih untuk menggunakan algoritma yang sama atau yang

baru, Amazon S3 menghitung nilai checksum baru setelah objek disalin. Amazon S3 tidak secara langsung menyalin nilai checksum. Nilai checksum objek yang dimuat dengan menggunakan unggahan multipart mungkin berubah. Untuk informasi selengkapnya tentang bagaimana checksum dihitung, lihat Menggunakan checksum tingkat bagian untuk unggahan multibagian.

Menyalin beberapa objek dalam satu permintaan

Untuk menyalin lebih dari satu objek Amazon S3 dengan satu permintaan, Anda juga dapat menggunakan Operasi Batch S3. Anda menyediakan daftar objek yang akan dioperasikan kepada Operasi Batch S3. Operasi Batch S3 akan memanggil masing-masing operasi API untuk melakukan operasi tertentu. Satu tugas Operasi Batch dapat melakukan operasi tertentu pada miliaran objek yang berisi data sebesar eksabita.

Operasi Batch S3 memiliki fitur melacak progres, mengirimkan notifikasi, dan menyimpan laporan penyelesaian terperinci dari semua tindakan, menyediakan pengalaman yang dikelola sepenuhnya, dapat diaudit, dan nirserver. Anda dapat menggunakan Operasi Batch S3 melalui konsol Amazon S3, SDK AWS CLI AWS , atau REST API. Untuk informasi selengkapnya, lihat the section called “Dasar-dasar Operasi Batch”.

Menyalin objek ke ember direktori

Untuk informasi tentang menyalin objek ke bucket direktori, lihatMenyalin objek ke bucket direktori. Untuk informasi tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat dan. Apa itu S3 Express One Zone? Ember direktori

Untuk menyalin objek

Untuk menyalin objek, gunakan metode berikut.

Menggunakan konsol S3

> **Note**
>
> - Saat menyalin objek dengan menggunakan konsol Amazon S3, Anda harus memiliki `s3:ListAllMyBuckets` izin. Konsol memerlukan izin ini untuk memvalidasi operasi Salin. Misalnya kebijakan yang memberikan izin ini, lihatthe section called “Contoh kebijakan berbasis identitas”.
>
> Jika Anda menyalin objek yang memiliki tag yang ditentukan pengguna, Anda juga harus memiliki izin. `s3:GetObjectTagging` Jika Anda menyalin objek yang tidak memiliki tag

yang ditentukan pengguna tetapi berukuran lebih dari 16 MB, Anda juga harus memiliki izin s3:. `GetObjectTagging`

Jika kebijakan bucket tujuan menolak `s3:GetObjectTagging` tindakan, objek akan disalin tanpa tag yang ditentukan pengguna, dan Anda akan menerima kesalahan.

- Objek yang dienkripsi dengan kunci enkripsi yang disediakan pelanggan (SSE-C) tidak dapat disalin dengan menggunakan konsol S3. Untuk menyalin objek yang dienkripsi dengan SSE-C, gunakan, AWS SDK AWS CLI, atau Amazon S3 REST API.
- Penyalinan objek Lintas Wilayah yang dienkripsi dengan SSE-KMS tidak didukung oleh konsol Amazon S3. Untuk menyalin objek yang dienkripsi dengan SSE-KMS di seluruh Wilayah, gunakan, AWS SDK AWS CLI, atau Amazon S3 REST API.

Untuk menyalin objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket, lalu pilih tab Bucket tujuan umum. Navigasikan ke bucket atau folder Amazon S3 yang berisi objek yang ingin Anda salin.
3. Centang kotak di sebelah kiri nama objek yang ingin Anda salin.
4. Pada menu Tindakan, pilih Salin dari daftar opsi yang muncul.
5. Pilih jenis tujuan dan akun tujuan. Untuk menentukan jalur tujuan, pilih Jelajahi S3, navigasi ke tujuan, dan pilih kotak centang di sebelah kiri tujuan. Pilih Pilih tujuan di sudut kanan bawah.

   Atau, masukkan jalur tujuan.
6. Jika Anda tidak mengaktifkan Penentuan Versi bucket, Anda mungkin diminta untuk menyatakan bahwa objek yang ada dengan nama yang sama akan ditimpa. Jika setuju, pilih kotak centang dan lanjutkan. Jika Anda ingin menyimpan semua versi objek dalam bucket ini, pilih Aktifkan Penentuan Versi Bucket. Anda juga dapat memperbarui enkripsi default dan properti Kunci Objek S3.
7. Di bawah Checksum tambahan, pilih apakah Anda ingin menyalin objek menggunakan fungsi checksum yang sudah ada, atau mengubah fungsi checksum yang sudah ada dengan yang baru. Saat Anda mengunggah objek, Anda memiliki opsi untuk menentukan algoritma checksum yang digunakan untuk memverifikasi integritas data. Saat menyalin objek, Anda memiliki opsi untuk memilih fungsi baru. Jika Anda awalnya tidak menentukan checksum tambahan, Anda dapat menggunakan bagian opsi salin ini untuk menambahkannya.

> **Note**
>
> Bahkan jika Anda memilih untuk menggunakan fungsi checksum yang sama, nilai checksum Anda mungkin berubah jika Anda menyalin objek dan ukurannya lebih dari 16 MB. Nilai checksum mungkin berubah karena cara checksum dihitung untuk unggahan multibagian. Untuk informasi selengkapnya tentang perubahan checksum saat menyalin objek, lihat Menggunakan checksum tingkat bagian untuk unggahan multibagian.

Untuk mengubah fungsi checksum, pilih Ganti dengan fungsi checksum baru. Pilih fungsi checksum baru dari kotak. Ketika objek disalin, checksum baru dihitung dan disimpan menggunakan algoritma yang ditentukan.

8. Pilih Salin di sudut kanan bawah. Amazon S3 menyalin objek Anda ke tujuan.

Menggunakan AWS SDK

Contoh di bagian ini memperlihatkan cara untuk menyalin objek hingga 5 GB dalam satu operasi. Untuk menyalin objek yang lebih besar dari 5 GB, Anda harus menggunakan unggahan multibagian. Untuk informasi selengkapnya, lihat Menyalin objek menggunakan unggahan multibagian.

Java

Example

Contoh berikut menyalin objek di Amazon S3 menggunakan AWS SDK for Java. Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CopyObjectRequest;

import java.io.IOException;
```

```
public class CopyObjectSingleOperation {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String sourceKey = "*** Source object key *** ";
        String destinationKey = "*** Destination object key ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Copy the object into a new object in the same bucket.
            CopyObjectRequest copyObjRequest = new CopyObjectRequest(bucketName,
 sourceKey, bucketName, destinationKey);
            s3Client.copyObject(copyObjRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Contoh C # berikut menggunakan tingkat tinggi AWS SDK for .NET untuk menyalin objek yang sebesar 5 GB dalam satu operasi. Untuk objek yang lebih besar dari 5 GB, gunakan contoh salinan unggahan multibagian yang dijelaskan dalam Menyalin objek menggunakan unggahan multibagian.

Contoh ini menampilkan pembuatan salinan objek yang berukuran maksimum sebesar 5 GB. Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
```

```
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CopyObjectTest
    {
        private const string sourceBucket = "*** provide the name of the bucket with
 source object ***";
        private const string destinationBucket = "*** provide the name of the bucket
 to copy the object to ***";
        private const string objectKey = "*** provide the name of object to copy
 ***";
        private const string destObjectKey = "*** provide the destination object key
 name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            Console.WriteLine("Copying an object");
            CopyingObjectAsync().Wait();
        }

        private static async Task CopyingObjectAsync()
        {
            try
            {
                CopyObjectRequest request = new CopyObjectRequest
                {
                    SourceBucket = sourceBucket,
                    SourceKey = objectKey,
                    DestinationBucket = destinationBucket,
                    DestinationKey = destObjectKey
                };
                CopyObjectResponse response = await
 s3Client.CopyObjectAsync(request);
            }
            catch (AmazonS3Exception e)
```

```
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

PHP

Topik ini memandu Anda menggunakan kelas dari versi 3 AWS SDK for PHP untuk menyalin satu objek dan beberapa objek dalam Amazon S3, dari satu bucket ke bucket lainnya atau dalam bucket yang sama.

Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

Contoh PHP berikut menggambarkan penggunaan `copyObject()` metode untuk menyalin satu objek dalam Amazon S3. Hal ini juga menunjukkan bagaimana untuk membuat beberapa salinan dari objek dengan menggunakan batch panggilan ke `CopyObject` dengan `getcommand()` metode.

Menyalin objek

1	Buat instans dari klien Amazon S3 dengan menggunakan konstruktor kelas `Aws\S3\S3Client` .
2	Untuk membuat beberapa salinan objek, Anda menjalankan sekumpulan panggilan ke getCommand()metode klien Amazon S3, yang diwarisi dari kelas. Aws\CommandInterface Anda menyediakan perintah `CopyObject` sebagai argumen pertama dan array yang berisi bucket sumber, nama kunci sumber, bucket target, dan nama kunci target sebagai argumen kedua.

```
require 'vendor/autoload.php';
```

```
use Aws\CommandPool;
use Aws\Exception\AwsException;
use Aws\ResultInterface;
use Aws\S3\S3Client;

$sourceBucket = '*** Your Source Bucket Name ***';
$sourceKeyname = '*** Your Source Object Key ***';
$targetBucket = '*** Your Target Bucket Name ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region' => 'us-east-1'
]);

// Copy an object.
$s3->copyObject([
    'Bucket' => $targetBucket,
    'Key' => "$sourceKeyname-copy",
    'CopySource' => "$sourceBucket/$sourceKeyname",
]);

// Perform a batch of CopyObject operations.
$batch = array();
for ($i = 1; $i <= 3; $i++) {
    $batch[] = $s3->getCommand('CopyObject', [
        'Bucket' => $targetBucket,
        'Key' => "{targetKeyname}-$i",
        'CopySource' => "$sourceBucket/$sourceKeyname",
    ]);
}
try {
    $results = CommandPool::batch($s3, $batch);
    foreach ($results as $result) {
        if ($result instanceof ResultInterface) {
            // Result handling here
        }
        if ($result instanceof AwsException) {
            // AwsException handling here
        }
    }
} catch (Exception $e) {
    // General error handling here
}
```

Python

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource in
 Boto3
                            that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key
```

```
    def copy(self, dest_object):
        """
        Copies the object to another bucket.

        :param dest_object: The destination object initialized with a bucket and
 key.
                              This is a Boto3 Object resource.
        """
        try:
            dest_object.copy_from(
                CopySource={"Bucket": self.object.bucket_name, "Key":
 self.object.key}
            )
            dest_object.wait_until_exists()
            logger.info(
                "Copied object from %s:%s to %s:%s.",
                self.object.bucket_name,
                self.object.key,
                dest_object.bucket_name,
                dest_object.key,
            )
        except ClientError:
            logger.exception(
                "Couldn't copy object from %s/%s to %s/%s.",
                self.object.bucket_name,
                self.object.key,
```

```
                dest_object.bucket_name,
                dest_object.key,
            )
            raise
```

Ruby

Tugas berikut memandu Anda menggunakan Ruby class untuk menyalin objek di Amazon S3 dari satu bucket ke bucket lainnya atau dalam bucket yang sama.

Menyalin objek

1	Gunakan permata termodulasi Amazon S3 untuk versi 3 dari, require AWS SDK for Rubyaws-sdk-s3 , dan berikan kredensyal Anda. AWS Untuk informasi lebih lanjut tentang cara menyediakan kredensial Anda, lihat Membuat permintaan menggunakan Akun AWS atau kredensil pengguna IAM.
2	Berikan informasi permintaan, seperti nama bucket sumber, nama kunci sumber, nama bucket tujuan, dan kunci tujuan.

Contoh Ruby kode berikut menunjukkan tugas-tugas sebelumnya dengan menggunakan #copy_object metode untuk menyalin objek dari satu ember ke ember lainnya.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectCopyWrapper
  attr_reader :source_object

  # @param source_object [Aws::S3::Object] An existing Amazon S3 object. This is
 used as the source object for
  #                                       copy actions.
  def initialize(source_object)
    @source_object = source_object
  end

  # Copy the source object to the specified target bucket and rename it with the
 target key.
  #
```

```
  # @param target_bucket [Aws::S3::Bucket] An existing Amazon S3 bucket where the
 object is copied.
  # @param target_object_key [String] The key to give the copy of the object.
  # @return [Aws::S3::Object, nil] The copied object when successful; otherwise,
 nil.
  def copy_object(target_bucket, target_object_key)
    @source_object.copy_to(bucket: target_bucket.name, key: target_object_key)
    target_bucket.object(target_object_key)
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't copy #{@source_object.key} to #{target_object_key}. Here's why:
 #{e.message}"
  end
end

# Example usage:
def run_demo
  source_bucket_name = "doc-example-bucket1"
  source_key = "my-source-file.txt"
  target_bucket_name = "doc-example-bucket2"
  target_key = "my-target-file.txt"

  source_bucket = Aws::S3::Bucket.new(source_bucket_name)
  wrapper = ObjectCopyWrapper.new(source_bucket.object(source_key))
  target_bucket = Aws::S3::Bucket.new(target_bucket_name)
  target_object = wrapper.copy_object(target_bucket, target_key)
  return unless target_object

  puts "Copied #{source_key} from #{source_bucket_name} to
 #{target_object.bucket_name}:#{target_object.key}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Penggunaan API REST

Contoh ini menjelaskan cara menyalin objek dengan menggunakan Amazon S3 REST API. Untuk informasi selengkapnya tentang API REST, lihat CopyObject.

Contoh ini menyalin objek `flotsam` dari bucket *`example-s3-bucket1`* ke objek `jetsam` dari bucket *`example-s3-bucket2`*, dengan mempertahankan metadatanya.

```
PUT /jetsam HTTP/1.1
```

```
Host: example-s3-bucket2.s3.amazonaws.com
x-amz-copy-source: /example-s3-bucket1/flotsam
Authorization: AWS AKIAIOSFODNN7EXAMPLE:ENoSbxYByFA0UGLZUqJN5EUnLDg=
Date: Wed, 20 Feb 2008 22:12:21 +0000
```

Tanda tangannya dihasilkan dari informasi berikut.

```
PUT\r\n
\r\n
\r\n
Wed, 20 Feb 2008 22:12:21 +0000\r\n

x-amz-copy-source:/example-s3-bucket1/flotsam\r\n
/example-s3-bucket2/jetsam
```

Amazon S3 akan menampilkan respons berikut, yang menentukan ETag dari objek dan kapan terakhir kali objek tersebut diubah.

```
HTTP/1.1 200 OK
x-amz-id-2: Vyaxt7qEbzv34BnSu5hctyyNSlHTYZFMWK4FtzO+iX8JQNyaLdTshL0KxatbaOZt
x-amz-request-id: 6B13C3C5B34AF333
Date: Wed, 20 Feb 2008 22:13:01 +0000

Content-Type: application/xml
Transfer-Encoding: chunked
Connection: close
Server: AmazonS3
<?xml version="1.0" encoding="UTF-8"?>

<CopyObjectResult>
   <LastModified>2008-02-20T22:13:01</LastModified>
   <ETag>"7e9c608af58950deeb370c98608ed097"</ETag>
</CopyObjectResult>
```

Menggunakan AWS CLI

Anda juga dapat menggunakan AWS Command Line Interface (AWS CLI) untuk menyalin objek S3. Untuk informasi selengkapnya, lihat copy-object dalam AWS CLI Referensi Perintah.

Untuk informasi tentang AWS CLI, lihat Apa itu AWS Command Line Interface? dalam AWS Command Line Interface User Guide.

Untuk memindahkan objek

Untuk memindahkan objek, gunakan metode berikut.

Menggunakan konsol S3

> **Note**
>
> - Jika Anda memindahkan objek yang memiliki tag yang ditentukan pengguna, Anda harus memiliki izin. `s3:GetObjectTagging` Jika Anda memindahkan objek yang tidak memiliki tag yang ditentukan pengguna tetapi berukuran lebih dari 16 MB, Anda juga harus memiliki izin s3:. `GetObjectTagging`
>
  Jika kebijakan bucket tujuan menolak `s3:GetObjectTagging` tindakan, objek akan dipindahkan tanpa tag yang ditentukan pengguna, dan Anda akan menerima kesalahan.
> - Objek yang dienkripsi dengan kunci enkripsi yang disediakan pelanggan (SSE-C) tidak dapat dipindahkan dengan menggunakan konsol Amazon S3. Untuk memindahkan objek yang dienkripsi dengan SSE-C, gunakan, AWS SDK AWS CLI, atau Amazon S3 REST API.
> - Saat memindahkan folder, tunggu operasi Pindahkan selesai sebelum membuat perubahan tambahan di folder.
> - Anda tidak dapat menggunakan alias titik akses S3 sebagai sumber atau tujuan untuk operasi Pindah di konsol Amazon S3.

Untuk memindahkan objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket, lalu pilih tab Bucket tujuan umum. Navigasikan ke bucket atau folder Amazon S3 yang berisi objek yang ingin Anda pindahkan.
3. Centang kotak di sebelah kiri nama objek yang ingin Anda pindahkan.
4. Pada menu Tindakan, pilih Pindahkan.
5. Untuk menentukan jalur tujuan, pilih Jelajahi S3, navigasi ke tujuan, dan pilih kotak centang di sebelah kiri tujuan. Pilih Pilih tujuan di sudut kanan bawah.

   Atau, masukkan jalur tujuan.

6. Jika Anda tidak mengaktifkan Penentuan Versi bucket, Anda mungkin diminta untuk menyatakan bahwa objek yang ada dengan nama yang sama akan ditimpa. Jika setuju, pilih kotak centang dan lanjutkan. Jika Anda ingin menyimpan semua versi objek dalam bucket ini, pilih Aktifkan Penentuan Versi Bucket. Anda juga dapat memperbarui enkripsi default dan properti Kunci Objek.
7. Pilih Pindahkan di sudut kanan bawah. Amazon S3 memindahkan objek Anda ke tujuan.

> **Note**
>
> - Tindakan ini membuat salinan semua objek tertentu dengan pengaturan yang diperbarui, memperbarui tanggal modifikasi terakhir di lokasi tertentu, dan menambahkan penanda hapus ke objek asli.
> - Tindakan ini memperbarui metadata untuk Penentuan Versi bucket, enkripsi, fitur Kunci Objek, dan objek yang diarsipkan.

Menggunakan AWS CLI

Anda juga dapat menggunakan AWS Command Line Interface (AWS CLI) untuk memindahkan objek S3. Untuk informasi selengkapnya, lihat mv dalam AWS CLI Referensi Perintah.

Untuk informasi tentang AWS CLI, lihat Apa itu AWS Command Line Interface? dalam AWS Command Line Interface User Guide.

Untuk mengganti nama objek

Untuk mengganti nama objek, gunakan prosedur berikut.

> **Note**
>
> - Mengganti nama objek membuat salinan objek dengan tanggal modifikasi terakhir baru, dan kemudian menambahkan penanda hapus ke objek asli.
> - Pengaturan bucket untuk enkripsi default diterapkan secara otomatis ke objek tertentu yang tidak dienkripsi.
> - Anda tidak dapat menggunakan konsol Amazon S3 untuk mengganti nama objek dengan kunci enkripsi yang disediakan pelanggan (SSE-C). Untuk mengganti nama objek yang

dienkripsi dengan SSE-C, gunakan, AWS SDK AWS CLI, atau Amazon S3 REST API untuk menyalin objek tersebut dengan nama baru.

- Jika bucket ini menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek S3, daftar kontrol akses objek (ACL) tidak akan disalin.
- Jika Anda mengganti nama objek yang memiliki tag yang ditentukan pengguna, Anda harus memiliki izin. `s3:GetObjectTagging` Jika Anda mengganti nama objek yang tidak memiliki tag yang ditentukan pengguna tetapi berukuran lebih dari 16 MB, Anda juga harus memiliki izin s3:. `GetObjectTagging`

  Jika kebijakan bucket tujuan menolak `s3:GetObjectTagging` tindakan, objek akan diganti namanya, tetapi tag yang ditentukan pengguna akan dihapus dari objek, dan Anda akan menerima kesalahan.

Untuk mengganti nama objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket, lalu pilih tab Bucket tujuan umum. Arahkan ke bucket Amazon S3 atau folder yang berisi objek yang ingin Anda ganti namanya.
3. Pilih kotak centang di sebelah kiri nama objek yang ingin Anda ganti namanya.
4. Pada menu Tindakan, pilih Ganti nama objek.
5. Di kotak Nama objek baru, masukkan nama baru untuk objek tersebut.
6. Pilih Simpan perubahan di sudut kanan bawah. Amazon S3 mengganti nama objek Anda.

Mengunggah objek

Bagian ini menjelaskan cara untuk mengunduh objek dari bucket Amazon S3. Dengan Amazon S3, Anda dapat menyimpan objek dalam satu atau lebih bucket, dan setiap objek dapat berukuran hingga 5 TB. Objek Amazon S3 apa pun yang tidak diarsipkan dapat diakses secara real time. Namun, objek yang diarsipkan harus dipulihkan sebelum dapat diunduh. Untuk informasi tentang objek yang diarsipkan, lihat the section called “Mengunduh objek yang diarsipkan”.

Anda dapat mengunduh satu objek dengan menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), AWS SDK, atau Amazon S3 REST API. Untuk mengunduh objek dari

S3 tanpa menulis kode atau menjalankan perintah apa pun, gunakan konsol S3. Untuk informasi selengkapnya, lihat the section called “Mengunduh objek”.

Untuk mengunduh beberapa objek AWS CloudShell, gunakan AWS CLI, atau AWS SDK. Untuk informasi selengkapnya, lihat the section called “Mengunduh beberapa objek”.

Jika Anda perlu mengunduh bagian dari suatu objek, Anda menggunakan parameter tambahan dengan AWS CLI atau REST API untuk menentukan hanya byte yang ingin Anda unduh. Untuk informasi selengkapnya, lihat the section called “Mengunduh bagian dari suatu objek”.

Jika Anda perlu mengunduh objek yang bukan milik Anda, mintalah pemilik objek untuk membuat URL yang telah ditentukan sebelumnya yang memungkinkan Anda mengunduh objek tersebut. Untuk informasi selengkapnya, lihat the section called “Mengunduh objek dari Akun AWS yang lain”.

Saat Anda mengunduh objek di luar AWS jaringan, biaya transfer data berlaku. Transfer data dalam AWS jaringan gratis dalam hal yang sama Wilayah AWS, tetapi Anda akan dikenakan biaya untuk `GET` permintaan apa pun. Untuk informasi selengkapnya tentang biaya transfer data dan pengambilan data, lihat Harga Amazon S3.

Topik

- Mengunduh objek
- Mengunduh beberapa objek
- Mengunduh bagian dari suatu objek
- Mengunduh objek dari Akun AWS yang lain
- Mengunduh objek yang diarsipkan
- Memecahkan masalah pengunduhan objek

Mengunduh objek

Anda dapat mengunduh objek menggunakan konsol Amazon S3, AWS SDK AWS CLI, atau REST API.

Menggunakan konsol S3

Bagian ini menjelaskan cara menggunakan konsol Amazon S3 untuk mengunduh objek dari bucket S3.

> **Note**
>
> - Anda hanya dapat mengunduh satu objek dalam satu waktu.
> - Jika Anda menggunakan konsol Amazon S3 untuk mengunduh objek yang nama kuncinya diakhiri dengan titik (`.`), titik tersebut dihapus dari nama kunci objek yang diunduh. Untuk mempertahankan periode di akhir nama objek yang diunduh, Anda harus menggunakan AWS Command Line Interface (AWS CLI), AWS SDK, atau Amazon S3 REST API.

Untuk mengunduh objek dari bucket S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Bucket, pilih nama bucket yang ingin Anda unduh objeknya.
3. Anda dapat mengunduh objek dari bucket S3 dengan cara berikut:
   - Pilih kotak centang di samping objek, dan pilih Unduh. Jika Anda ingin mengunduh objek ke folder tertentu, pada menu Tindakan, pilih Unduh sebagai.
   - Jika Anda ingin mengunduh versi objek tertentu, aktifkan Tampilkan versi (terletak di samping kotak pencarian). Centang kotak di samping versi objek yang Anda inginkan, dan pilih Unduh. Jika Anda ingin mengunduh objek ke folder tertentu, pada menu Tindakan, pilih Unduh sebagai.

Menggunakan AWS CLI

Contoh perintah `get-object` berikut menunjukkan bagaimana Anda dapat menggunakan AWS CLI untuk mengunduh objek dari Amazon S3. Perintah ini mendapatkan objek *`folder/my_image`* dari bucket *`example-s3-bucket1`*. Objek akan diunduh ke file yang bernama *`my_downloaded_image`*.

```
aws s3api get-object --bucket example-s3-bucket1 --key folder/
my_image my_downloaded_image
```

Untuk informasi dan contoh selengkapnya, lihat get-object di AWS CLI Referensi Perintah.

Menggunakan AWS SDK

Untuk contoh cara mengunduh objek dengan AWS SDK, lihatGunakan `GetObject` dengan AWS SDK atau CLI.

Untuk informasi umum tentang penggunaan AWS SDK yang berbeda, lihatMengembangkan dengan Amazon S3 menggunakan SDK AWS.

Penggunaan API REST

Anda dapat menggunakan API REST untuk mengambil objek dari Amazon S3. Untuk informasi selengkapnya, lihat GetObject dalam Referensi API Amazon Simple Storage Service.

Mengunduh beberapa objek

Anda dapat mengunduh beberapa objek dengan menggunakan AWS CloudShell, SDK AWS CLI, atau AWS SDK.

Menggunakan AWS CloudShell di AWS Management Console

AWS CloudShell adalah shell pra-otentikasi berbasis browser yang dapat Anda luncurkan langsung dari file. AWS Management Console

Untuk informasi lebih lanjut tentang AWS CloudShell, lihat Apa itu CloudShell? dalam AWS CloudShell User Guide.

> **Important**
>
> Dengan AWS CloudShell, direktori home Anda memiliki penyimpanan hingga 1GB per Wilayah AWS. Maka, Anda tidak dapat menyinkronkan bucket dengan objek yang berjumlah lebih dari jumlah ini. Untuk pembatasan yang lebih lengkap, lihat Kuota layanan dan batasan di AWS CloudShell Panduan Pengguna.

Untuk mengunduh objek dengan menggunakan AWS CloudShell

1. Masuk ke AWS Management Console dan buka CloudShell konsol di https://console.aws.amazon.com/cloudshell/.
2. Jalankan perintah berikut untuk menyinkronkan objek di bucket Anda CloudShell. Perintah berikut menyinkronkan objek dari bucket bernama *`example-s3-bucket1`* dan membuat

folder bernama*temp*. CloudShell CloudShell menyinkronkan objek Anda ke folder ini. Untuk menggunakan perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3 sync s3://example-s3-bucket1 ./temp
```

> **Note**
>
> Untuk melakukan pencocokan pola untuk mengecualikan atau menyertakan objek tertentu, Anda dapat menggunakan parameter `--exclude "value"` dan `--include "value"` dengan perintah `sync`.

3. Jalankan perintah berikut untuk meng-zip objek Anda di folder bernama *temp* ke file bernama *temp.zip*.

```
zip temp.zip -r temp/
```

4. Pilih Tindakan, lalu pilih Unduh file.
5. Masukkan nama file **`temp.zip`** lalu pilih Unduh.
6. (Opsional) Hapus *temp.zip* file dan objek yang disinkronkan ke *temp* folder di CloudShell. Dengan AWS CloudShell, Anda memiliki penyimpanan persisten hingga 1 GB untuk masing-masing Wilayah AWS.

   Anda dapat menggunakan contoh perintah berikut untuk menghapus file `.zip` dan folder Anda. Untuk menggunakan contoh perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
rm temp.zip && rm -rf temp/
```

Menggunakan AWS CLI

Contoh berikut menunjukkan bagaimana Anda dapat menggunakan AWS CLI untuk men-download semua file atau objek di bawah direktori tertentu atau awalan. Perintah ini menyalin semua objek dari bucket *example-s3-bucket1* ke direktori Anda saat ini. Untuk menggunakan perintah contoh ini, gunakan nama bucket Anda sebagai pengganti *example-s3-bucket1*.

```
aws s3 cp s3://example-s3-bucket1 . --recursive
```

Perintah berikut mengunduh semua objek di bawah prefiks *logs* di bucket *example-s3-bucket1* ke direktori Anda saat ini. Ini juga menggunakan parameter `--exclude` dan `--include` untuk menyalin hanya objek dengan sufiks *.log*. Untuk menggunakan contoh perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3 cp s3://example-s3-bucket1/logs/ . --recursive --exclude "*" --include "*.log"
```

Untuk informasi dan contoh selengkapnya, lihat cp di AWS CLI Referensi Perintah.

Menggunakan AWS SDK

Untuk contoh cara mengunduh semua objek di bucket Amazon S3 dengan AWS SDK, lihat. Mengunduh semua objek di bucket Amazon Simple Storage Service (Amazon S3) ke direktori lokal

Untuk informasi umum tentang penggunaan AWS SDK yang berbeda, lihatMengembangkan dengan Amazon S3 menggunakan SDK AWS.

Mengunduh bagian dari suatu objek

Anda dapat mengunduh bagian dari suatu objek dengan menggunakan AWS CLI atau REST API. Untuk melakukannya, Anda menggunakan parameter tambahan untuk menentukan bagian mana dari objek yang ingin Anda unduh.

Menggunakan AWS CLI

Perintah contoh berikut ini melakukan permintaan `GET` untuk rentang byte dalam objek bernama *folder/my_data* dalam bucket yang bernama *example-s3-bucket1*. Dalam permintaan tersebut, rentang byte harus diawali dengan `bytes=`. Objek parsial diunduh ke file output bernama *my_data_range*. Untuk menggunakan contoh perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3api get-object --bucket example-s3-bucket1 --key folder/my_data --range
 bytes=0-500 my_data_range
```

Untuk informasi dan contoh selengkapnya, lihat get-object di AWS CLI Referensi Perintah.

Untuk informasi selengkapnya tentang header `Range` HTTP, lihat RFC 9110 di situs web RFC Editor.

> **Note**
>
> Amazon S3 tidak mendukung pengambilan beberapa rentang data dalam satu permintaan `GET`.

Penggunaan API REST

Anda dapat menggunakan parameter `partNumber` dan `Range` di API REST untuk mengambil bagian objek dari Amazon S3. Untuk informasi selengkapnya, lihat GetObject dalam Referensi API Amazon Simple Storage Service.

Mengunduh objek dari Akun AWS yang lain

Anda dapat menggunakan URL yang telah ditandatangani sebelumnya untuk memberi orang lain akses yang dibatasi waktu ke objek Anda tanpa memperbarui kebijakan bucket Anda.

URL yang telah ditandatangani sebelumnya dapat dimasukkan di dalam browser, atau digunakan oleh program untuk mengunduh objek. Kredensi yang digunakan oleh URL adalah milik AWS pengguna yang membuat URL. Setelah URL dibuat, siapa pun yang memiliki URL yang telah ditentukan sebelumnya dapat mengunduh objek terkait hingga URL tersebut kedaluwarsa.

Menggunakan URL yang telah ditandatangani sebelumnya di konsol S3

Anda dapat menggunakan konsol Amazon S3 untuk menghasilkan URL yang telah ditandatangani untuk berbagi objek dengan mengikuti langkah-langkah berikut. Saat menggunakan konsol, waktu kedaluwarsa maksimum untuk URL yang telah ditandatangani sebelumnya adalah 12 jam dari waktu pembuatan.

Untuk menghasilkan URL yang telah ditandatangani sebelumnya menggunakan konsol Amazon S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di dalam daftar Bucket, pilih nama bucket yang berisi objek yang Anda inginkan sebagai URL yang telah ditandatangani.
4. Dalam daftar Objek, pilih objek yang ingin Anda buatkan URL yang telah ditandatangani.
5. Pada menu Tindakan objek, pilih Bagikan dengan URL yang telah ditandatangani.
6. Tentukan berapa lama Anda menginginkan URL yang telah ditandatangani tersebut valid.

7. Pilih Buat URL yang telah ditandatangani.
8. Ketika pesan konfirmasi muncul, URL secara otomatis disalin ke clipboard Anda. Anda akan melihat tombol untuk menyalin URL yang telah ditandatangani jika Anda perlu menyalinnya lagi.
9. Untuk mengunduh objek, tempel URL ke browser apa pun, dan objek akan mencoba mengunduh.

Untuk informasi selengkapnya tentang URL yang telah ditandatangani dan metode lain untuk membuatnya, lihat Bekerja dengan URL yang telah ditandatangani.

Mengunduh objek yang diarsipkan

Untuk mengurangi biaya penyimpanan objek yang jarang diakses, Anda dapat mengarsipkan objek tersebut. Ketika Anda mengarsipkan objek, objek tersebut dipindahkan ke penyimpanan berbiaya rendah, yang berarti Anda tidak dapat mengaksesnya secara real time. Untuk mengunduh objek yang diarsipkan, Anda harus memulihkannya terlebih dahulu.

Anda dapat memulihkan objek yang diarsipkan dalam hitungan menit atau jam, tergantung pada kelas penyimpanan. Anda dapat memulihkan objek yang diarsipkan menggunakan konsol Amazon S3, Operasi Batch S3, API REST Amazon S3, SDK, AWS dan (). AWS Command Line Interface AWS CLI

Untuk petunjuk, lihat Memulihkan objek yang diarsipkan. Setelah Anda memulihkan objek yang diarsipkan, Anda dapat mengunduhnya.

Memecahkan masalah pengunduhan objek

Izin yang tidak memadai atau kebijakan pengguna bucket atau AWS Identity and Access Management (IAM) yang salah dapat menyebabkan kesalahan saat Anda mencoba mengunduh objek dari Amazon S3. Masalah ini sering dapat menyebabkan kesalahan Akses Ditolak (403 Dilarang). di mana Amazon S3 tidak bisa memberikan akses ke sumber daya.

Untuk mengetahui penyebab umum kesalahan Akses Ditolak (403 Forbidden), lihat Pecahkan masalah kesalahan Akses Ditolak (403 Forbidden) di Amazon S3.

Memeriksa integritas objek

Amazon S3 menggunakan nilai checksum untuk memverifikasi integritas data yang Anda unggah atau unduh dari Amazon S3. Selain itu, Anda dapat meminta agar nilai checksum lain dihitung untuk

objek apa pun yang Anda simpan di Amazon S3. Anda dapat memilih dari salah satu dari beberapa algoritma checksum untuk digunakan saat mengunggah atau menyalin data Anda. Amazon S3 menggunakan algoritma ini untuk menghitung nilai checksum tambahan dan menyimpannya sebagai bagian dari metadata objek. Untuk mempelajari selengkapnya tentang cara menggunakan checksum tambahan untuk memverifikasi integritas data, lihat Tutorial: Memeriksa integritas data di Amazon S3 dengan checksum tambahan.

Saat Anda mengunggah objek, secara opsional Anda dapat menyertakan checksum yang telah dihitung sebelumnya sebagai bagian dari permintaan Anda. Amazon S3 membandingkan checksum yang disediakan dengan checksum yang dihitung dengan menggunakan algoritma yang Anda tentukan. Jika kedua nilainya tidak cocok, Amazon S3 menghasilkan kesalahan.

Menggunakan algoritma checksum yang didukung

Amazon S3 menawarkan Anda opsi untuk memilih algoritma checksum yang digunakan untuk memvalidasi data Anda selama pengunggahan atau pengunduhan. Anda dapat memilih salah satu algoritma pemeriksaan integritas data Secure Hash Algorithms (SHA) atau Cyclic Redundancy Check (CRC) berikut:

- CRC32
- CRC32C
- SHA-1
- SHA-256

Saat mengunggah sebuah objek, Anda dapat menentukan algoritma yang ingin Anda gunakan:

- Saat Anda menggunakan AWS Management Console, Anda memilih algoritma checksum yang ingin Anda gunakan. Ketika Anda melakukannya, Anda dapat menentukan nilai checksum objek secara opsional. Ketika Amazon S3 menerima objek, itu menghitung checksum dengan menggunakan algoritma yang Anda tentukan. Jika kedua nilai checksum tidak cocok, Amazon S3 menghasilkan kesalahan.
- Saat menggunakan suatu SDK, Anda dapat mengatur nilai parameter `x-amz-sdk-checksum-algorithm` ke algoritma yang ingin digunakan Amazon S3 saat menghitung checksum. Amazon S3 secara otomatis menghitung nilai checksum.
- Saat Anda menggunakan API REST, Anda tidak menggunakan parameter `x-amz-sdk-checksum-algorithm`. Sebagai gantinya, Anda menggunakan salah satu header kustom algoritma (misalnya, `x-amz-checksum-crc32`).

Untuk informasi selengkapnya tentang mengunggah objek, lihat Mengunggah Objek.

Untuk menerapkan nilai checksum ini ke objek yang sudah diunggah ke Amazon S3, Anda dapat menyalin objek tersebut. Saat Anda menyalin suatu objek, Anda dapat menentukan apakah Anda ingin menggunakan algoritma checksum yang ada atau menggunakan yang baru. Anda dapat menentukan algoritme checksum saat menggunakan mekanisme apa pun yang didukung untuk menyalin objek, termasuk Operasi Batch S3. Untuk informasi selengkapnya tentang Operasi Batch S3, lihat Melakukan operasi batch berskala besar pada objek Amazon S3.

> **Important**
>
> Jika Anda menggunakan unggahan multibagian dengan checksum tambahan, nomor bagian multibagian harus menggunakan nomor bagian berturut-turut. Saat menggunakan checksum tambahan, jika Anda mencoba menyelesaikan permintaan unggahan multibagian dengan nomor bagian yang tidak berurutan, Amazon S3 akan membuat kesalahan `500 Internal Server Error` HTTP.

Setelah mengunggah objek, Anda bisa mendapatkan nilai checksum dan membandingkannya dengan nilai checksum yang telah dihitung sebelumnya atau disimpan sebelumnya yang dihitung menggunakan algoritma yang sama.

Menggunakan konsol S3

Untuk mempelajari lebih lanjut tentang menggunakan konsol dan menentukan algoritma checksum yang akan digunakan saat mengunggah objek, lihat Mengunggah Objek dan Tutorial: Memeriksa integritas data di Amazon S3 dengan checksum tambahan.

Menggunakan AWS SDK

Contoh berikut menunjukkan bagaimana Anda dapat menggunakan AWS SDK untuk meng-upload file besar dengan upload multipart, men-download file besar, dan memvalidasi file upload multipart, semua dengan menggunakan SHA-256 untuk validasi file.

Java

Example Contoh: Mengunggah, mengunduh, dan memverifikasi file yang berukuran besar dengan SHA-256

Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import software.amazon.awssdk.auth.credentials.AwsCredentials;
import software.amazon.awssdk.auth.credentials.AwsCredentialsProvider;
import software.amazon.awssdk.core.ResponseInputStream;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.AbortMultipartUploadRequest;
import software.amazon.awssdk.services.s3.model.ChecksumAlgorithm;
import software.amazon.awssdk.services.s3.model.ChecksumMode;
import software.amazon.awssdk.services.s3.model.CompleteMultipartUploadRequest;
import software.amazon.awssdk.services.s3.model.CompleteMultipartUploadResponse;
import software.amazon.awssdk.services.s3.model.CompletedMultipartUpload;
import software.amazon.awssdk.services.s3.model.CompletedPart;
import software.amazon.awssdk.services.s3.model.CreateMultipartUploadRequest;
import software.amazon.awssdk.services.s3.model.CreateMultipartUploadResponse;
import software.amazon.awssdk.services.s3.model.GetObjectAttributesRequest;
import software.amazon.awssdk.services.s3.model.GetObjectAttributesResponse;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.s3.model.GetObjectTaggingRequest;
import software.amazon.awssdk.services.s3.model.ObjectAttributes;
import software.amazon.awssdk.services.s3.model.PutObjectTaggingRequest;
import software.amazon.awssdk.services.s3.model.Tag;
import software.amazon.awssdk.services.s3.model.Tagging;
import software.amazon.awssdk.services.s3.model.UploadPartRequest;
import software.amazon.awssdk.services.s3.model.UploadPartResponse;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.nio.ByteBuffer;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.Base64;
import java.util.List;

public class LargeObjectValidation {
    private static String FILE_NAME = "sample.file";
    private static String BUCKET = "sample-bucket";
```

```
        //Optional, if you want a method of storing the full multipart object
 checksum in S3.
        private static String CHECKSUM_TAG_KEYNAME = "fullObjectChecksum";
        //If you have existing full-object checksums that you need to validate
 against, you can do the full object validation on a sequential upload.
        private static String SHA256_FILE_BYTES = "htCM5g7ZNdoSw8bN/
mkgiAhXt5MFoVowVg+LE9aIQmI=";
        //Example Chunk Size - this must be greater than or equal to 5MB.
        private static int CHUNK_SIZE = 5 * 1024 * 1024;

        public static void main(String[] args) {
            S3Client s3Client = S3Client.builder()
                    .region(Region.US_EAST_1)
                    .credentialsProvider(new AwsCredentialsProvider() {
                        @Override
                        public AwsCredentials resolveCredentials() {
                            return new AwsCredentials() {
                                @Override
                                public String accessKeyId() {
                                    return Constants.ACCESS_KEY;
                                }

                                @Override
                                public String secretAccessKey() {
                                    return Constants.SECRET;
                                }
                            };
                        }
                    })
                    .build();
            uploadLargeFileBracketedByChecksum(s3Client);
            downloadLargeFileBracketedByChecksum(s3Client);
            validateExistingFileAgainstS3Checksum(s3Client);
        }

        public static void uploadLargeFileBracketedByChecksum(S3Client s3Client) {
            System.out.println("Starting uploading file validation");
            File file = new File(FILE_NAME);
            try (InputStream in = new FileInputStream(file)) {
                MessageDigest sha256 = MessageDigest.getInstance("SHA-256");
                CreateMultipartUploadRequest createMultipartUploadRequest =
 CreateMultipartUploadRequest.builder()
                        .bucket(BUCKET)
                        .key(FILE_NAME)
```

```
                        .checksumAlgorithm(ChecksumAlgorithm.SHA256)
                        .build();
                CreateMultipartUploadResponse createdUpload =
s3Client.createMultipartUpload(createMultipartUploadRequest);
                List<CompletedPart> completedParts = new ArrayList<CompletedPart>();
                int partNumber = 1;
                byte[] buffer = new byte[CHUNK_SIZE];
                int read = in.read(buffer);
                while (read != -1) {
                    UploadPartRequest uploadPartRequest =
UploadPartRequest.builder()

.partNumber(partNumber).uploadId(createdUpload.uploadId()).key(FILE_NAME).bucket(BUCKET).ch
                    UploadPartResponse uploadedPart =
s3Client.uploadPart(uploadPartRequest,
RequestBody.fromByteBuffer(ByteBuffer.wrap(buffer, 0, read)));
                    CompletedPart part =
CompletedPart.builder().partNumber(partNumber).checksumSHA256(uploadedPart.checksumSHA256()
                    completedParts.add(part);
                    sha256.update(buffer, 0, read);
                    read = in.read(buffer);
                    partNumber++;
                }
                String fullObjectChecksum =
Base64.getEncoder().encodeToString(sha256.digest());
                if (!fullObjectChecksum.equals(SHA256_FILE_BYTES)) {
                    //Because the SHA256 is uploaded after the part is uploaded; the
upload is bracketed and the full object can be fully validated.

s3Client.abortMultipartUpload(AbortMultipartUploadRequest.builder().bucket(BUCKET).key(FILE
                    throw new IOException("Byte mismatch between stored checksum and
upload, do not proceed with upload and cleanup");
                }
                CompletedMultipartUpload completedMultipartUpload =
CompletedMultipartUpload.builder().parts(completedParts).build();
                CompleteMultipartUploadResponse completedUploadResponse =
s3Client.completeMultipartUpload(

CompleteMultipartUploadRequest.builder().bucket(BUCKET).key(FILE_NAME).uploadId(createdUplo
                Tag checksumTag =
Tag.builder().key(CHECKSUM_TAG_KEYNAME).value(fullObjectChecksum).build();
                //Optionally, if you need the full object checksum stored with the
file; you could add it as a tag after completion.
```

```
s3Client.putObjectTagging(PutObjectTaggingRequest.builder().bucket(BUCKET).key(FILE_NAME).t
            } catch (IOException | NoSuchAlgorithmException e) {
                e.printStackTrace();
            }
            GetObjectAttributesResponse
                    objectAttributes =
s3Client.getObjectAttributes(GetObjectAttributesRequest.builder().bucket(BUCKET).key(FILE_N
                    .objectAttributes(ObjectAttributes.OBJECT_PARTS,
ObjectAttributes.CHECKSUM).build());
            System.out.println(objectAttributes.objectParts().parts());
            System.out.println(objectAttributes.checksum().checksumSHA256());
        }

        public static void downloadLargeFileBracketedByChecksum(S3Client s3Client) {
            System.out.println("Starting downloading file validation");
            File file = new File("DOWNLOADED_" + FILE_NAME);
            try (OutputStream out = new FileOutputStream(file)) {
                GetObjectAttributesResponse
                        objectAttributes =
s3Client.getObjectAttributes(GetObjectAttributesRequest.builder().bucket(BUCKET).key(FILE_N
                        .objectAttributes(ObjectAttributes.OBJECT_PARTS,
ObjectAttributes.CHECKSUM).build());
                //Optionally if you need the full object checksum, you can grab a
tag you added on the upload
                List<Tag> objectTags =
s3Client.getObjectTagging(GetObjectTaggingRequest.builder().bucket(BUCKET).key(FILE_NAME).b
                String fullObjectChecksum = null;
                for (Tag objectTag : objectTags) {
                    if (objectTag.key().equals(CHECKSUM_TAG_KEYNAME)) {
                        fullObjectChecksum = objectTag.value();
                        break;
                    }
                }
                MessageDigest sha256FullObject =
MessageDigest.getInstance("SHA-256");
                MessageDigest sha256ChecksumOfChecksums =
MessageDigest.getInstance("SHA-256");

                //If you retrieve the object in parts, and set the ChecksumMode to
enabled, the SDK will automatically validate the part checksum
                for (int partNumber = 1; partNumber <=
objectAttributes.objectParts().totalPartsCount(); partNumber++) {
                    MessageDigest sha256Part = MessageDigest.getInstance("SHA-256");
```

```
                    ResponseInputStream<GetObjectResponse> response =
 s3Client.getObject(GetObjectRequest.builder().bucket(BUCKET).key(FILE_NAME).partNumber(part
                    GetObjectResponse getObjectResponse = response.response();
                    byte[] buffer = new byte[CHUNK_SIZE];
                    int read = response.read(buffer);
                    while (read != -1) {
                        out.write(buffer, 0, read);
                        sha256FullObject.update(buffer, 0, read);
                        sha256Part.update(buffer, 0, read);
                        read = response.read(buffer);
                    }
                    byte[] sha256PartBytes = sha256Part.digest();
                    sha256ChecksumOfChecksums.update(sha256PartBytes);
                    //Optionally, you can do an additional manual validation again
 the part checksum if needed in addition to the SDK check
                    String base64PartChecksum =
 Base64.getEncoder().encodeToString(sha256PartBytes);
                    String base64PartChecksumFromObjectAttributes =
 objectAttributes.objectParts().parts().get(partNumber - 1).checksumSHA256();
                    if (!
base64PartChecksum.equals(getObjectResponse.checksumSHA256()) || !
base64PartChecksum.equals(base64PartChecksumFromObjectAttributes)) {
                        throw new IOException("Part checksum didn't match for the
 part");
                    }
                    System.out.println(partNumber + " " + base64PartChecksum);
                }
                //Before finalizing, do the final checksum validation.
                String base64FullObject =
 Base64.getEncoder().encodeToString(sha256FullObject.digest());
                String base64ChecksumOfChecksums =
 Base64.getEncoder().encodeToString(sha256ChecksumOfChecksums.digest());
                if (fullObjectChecksum != null && !
fullObjectChecksum.equals(base64FullObject)) {
                    throw new IOException("Failed checksum validation for full
 object");
                }
                System.out.println(fullObjectChecksum);
                String base64ChecksumOfChecksumFromAttributes =
 objectAttributes.checksum().checksumSHA256();
                if (base64ChecksumOfChecksumFromAttributes != null && !
base64ChecksumOfChecksums.equals(base64ChecksumOfChecksumFromAttributes)) {
                    throw new IOException("Failed checksum validation for full
 object checksum of checksums");
```

```
                }
                System.out.println(base64ChecksumOfChecksumFromAttributes);
                out.flush();
            } catch (IOException | NoSuchAlgorithmException e) {
                //Cleanup bad file
                file.delete();
                e.printStackTrace();
            }
        }

        public static void validateExistingFileAgainstS3Checksum(S3Client s3Client)
{
            System.out.println("Starting existing file validation");
            File file = new File("DOWNLOADED_" + FILE_NAME);
            GetObjectAttributesResponse
                    objectAttributes =
s3Client.getObjectAttributes(GetObjectAttributesRequest.builder().bucket(BUCKET).key(FILE_N
                    .objectAttributes(ObjectAttributes.OBJECT_PARTS,
ObjectAttributes.CHECKSUM).build());
            try (InputStream in = new FileInputStream(file)) {
                MessageDigest sha256ChecksumOfChecksums =
MessageDigest.getInstance("SHA-256");
                MessageDigest sha256Part = MessageDigest.getInstance("SHA-256");
                byte[] buffer = new byte[CHUNK_SIZE];
                int currentPart = 0;
                int partBreak =
objectAttributes.objectParts().parts().get(currentPart).size();
                int totalRead = 0;
                int read = in.read(buffer);
                while (read != -1) {
                    totalRead += read;
                    if (totalRead >= partBreak) {
                        int difference = totalRead - partBreak;
                        byte[] partChecksum;
                        if (totalRead != partBreak) {
                            sha256Part.update(buffer, 0, read - difference);
                            partChecksum = sha256Part.digest();
                            sha256ChecksumOfChecksums.update(partChecksum);
                            sha256Part.reset();
                            sha256Part.update(buffer, read - difference,
difference);
                        } else {
                            sha256Part.update(buffer, 0, read);
                            partChecksum = sha256Part.digest();
```

```
                            sha256ChecksumOfChecksums.update(partChecksum);
                            sha256Part.reset();
                        }
                        String base64PartChecksum =
 Base64.getEncoder().encodeToString(partChecksum);
                        if (!
base64PartChecksum.equals(objectAttributes.objectParts().parts().get(currentPart).checksumSH
 {
                            throw new IOException("Part checksum didn't match S3");
                        }
                        currentPart++;
                        System.out.println(currentPart + " " + base64PartChecksum);
                        if (currentPart <
 objectAttributes.objectParts().totalPartsCount()) {
                            partBreak +=
 objectAttributes.objectParts().parts().get(currentPart - 1).size();
                        }
                    } else {
                        sha256Part.update(buffer, 0, read);
                    }
                    read = in.read(buffer);
                }
                if (currentPart != objectAttributes.objectParts().totalPartsCount())
 {
                    currentPart++;
                    byte[] partChecksum = sha256Part.digest();
                    sha256ChecksumOfChecksums.update(partChecksum);
                    String base64PartChecksum =
 Base64.getEncoder().encodeToString(partChecksum);
                    System.out.println(currentPart + " " + base64PartChecksum);
                }

                String base64CalculatedChecksumOfChecksums =
 Base64.getEncoder().encodeToString(sha256ChecksumOfChecksums.digest());
                System.out.println(base64CalculatedChecksumOfChecksums);
                System.out.println(objectAttributes.checksum().checksumSHA256());
                if (!
base64CalculatedChecksumOfChecksums.equals(objectAttributes.checksum().checksumSHA256()))
 {
                    throw new IOException("Full object checksum of checksums don't
 match S3");
                }

            } catch (IOException | NoSuchAlgorithmException e) {
```

```
                e.printStackTrace();
            }
        }
    }
```

Penggunaan API REST

Anda dapat mengirim permintaan REST untuk mengunggah objek dengan nilai checksum untuk memverifikasi integritas data dengan PutObject. Anda juga dapat mengambil nilai checksum untuk objek yang menggunakan GetObjectatau. HeadObject

Menggunakan AWS CLI

Anda dapat mengirim permintaan PUT untuk mengunggah objek hingga 5 GB dalam satu operasi. Untuk informasi selengkapnya, lihat `PutObject` di Referensi Perintah AWS CLI . Anda juga dapat menggunakan `get-object` dan `head-object` untuk mengambil checksum dari objek yang sudah diunggah guna memverifikasi integritas data.

Untuk selengkapnya, lihat FAQ Amazon S3 CLI di Panduan Pengguna.AWS Command Line Interface

Menggunakan Content-MD5 saat mengunggah objek

Cara lain untuk memverifikasi integritas objek Anda setelah diunggah adalah dengan memberikan intisari MD5 objek saat Anda mengunggahnya. Jika Anda menghitung intisari MD5 untuk objek Anda, Anda dapat memberikan intisari dengan perintah PUT menggunakan header `Content-MD5`.

Setelah mengunggah objek, Amazon S3 menghitung intisari MD5 objek dan membandingkannya dengan nilai yang Anda berikan. Permintaan hanya berhasil jika kedua intisari cocok.

Menyediakan intisari MD5 tidak diperlukan, tetapi Anda dapat menggunakannya untuk memverifikasi integritas objek sebagai bagian dari proses pengunggahan.

Menggunakan Content-MD5 dan ETag untuk memverifikasi objek yang diunggah

Tag entitas (ETag) untuk sebuah objek mewakili versi objek tertentu. Perlu diingat bahwa ETag mencerminkan perubahan hanya pada konten suatu objek, bukan metadatanya. Jika hanya metadata objek yang berubah, maka ETag-nya tetap sama.

Tergantung pada objeknya, ETag dari objek mungkin rangkuman MD5 dari data objek:

- Jika suatu objek dibuat melalui operasi `PutObject`, `PostObject`, atau `CopyObject`, atau melalui AWS Management Console, dan objek tersebut juga berupa teks biasa atau dienkripsi dengan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3), objek tersebut memiliki ETag yang merupakan intisari MD5 dari data objeknya.
- Jika objek dibuat oleh,, atau `CopyObject` operasi`PutObject`, atau melalui `PostObject` AWS Management Console, dan objek itu dienkripsi oleh enkripsi sisi server dengan kunci yang disediakan pelanggan (SSE-C) atau enkripsi sisi server dengan () kunci (SSE-KMS), objek tersebut memiliki ETag AWS Key Management Service yang bukan merupakan intisari MD5 dari data objeknya.AWS KMS
- Jika sebuah objek dibuat oleh operasi `Multipart Upload` atau `Part Copy`, ETag objek bukanlah intisari MD5, terlepas dari metode enkripsinya. Jika objek lebih besar dari 16 MB, mengunggah atau menyalin AWS Management Console dari objek tersebut sebagai unggahan multibagian, dan oleh karena itu ETag bukan merupakan intisari MD5.

Untuk objek di mana ETag adalah intisari `Content-MD5` objek, Anda dapat membandingkan nilai ETag objek dengan intisari `Content-MD5` yang dihitung atau disimpan sebelumnya.

Menggunakan checksum trailing

Saat mengunggah objek ke Amazon S3, Anda dapat memberikan checksum yang telah dihitung sebelumnya untuk objek tersebut atau menggunakan SDK untuk secara otomatis membuat checksum AWS tambahan atas nama Anda. Jika Anda memutuskan untuk menggunakan checksum trailing, Amazon S3 secara otomatis menghasilkan checksum dengan menggunakan algoritma yang Anda tentukan dan menggunakannya untuk memvalidasi integritas objek selama pengunggahan.

Untuk membuat checksum tambahan saat menggunakan AWS SDK, isi `ChecksumAlgorithm` parameter dengan algoritme pilihan Anda. SDK menggunakan algoritma tersebut untuk menghitung checksum untuk objek Anda (atau bagian objek), dan secara otomatis menambahkannya ke akhir permintaan unggahan Anda. Perilaku ini menghemat waktu Anda, sebab Amazon S3 melakukan verifikasi dan pengunggahan data Anda dalam sekali jalan.

> **Important**
>
> Jika Anda menggunakan S3 Lambda Objek, semua permintaan ke S3 Lambda Objek ditandatangani menggunakan, `s3-object-lambda` dan bukan `s3`. Perilaku ini memengaruhi tanda tangan nilai checksum trailing. Untuk informasi selengkapnya tentang S3 Lambda Objek, lihat Mengubah objek dengan S3 Lambda Objek.

Menggunakan checksum tingkat bagian untuk unggahan multibagian

Ketika objek diunggah ke Amazon S3, mereka dapat diunggah sebagai objek tunggal atau melalui proses pengunggahan multibagian. Objek yang lebih besar dari 16 MB dan yang diunggah melalui konsol akan secara otomatis diunggah menggunakan unggahan multibagian. Untuk informasi lebih lanjut tentang unggahan multibagian, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.

Ketika sebuah objek diunggah sebagai unggahan multibagian, ETag untuk objek bukanlah intisari MD5 dari seluruh objek. Amazon S3 menghitung intisari MD5 dari setiap bagian saat diunggah. Intisari MD5 digunakan untuk menentukan ETag untuk objek akhir. Amazon S3 menggabungkan byte untuk intisari MD5 dan kemudian menghitung intisari MD5 dari nilai-nilai gabungan ini. Langkah terakhir untuk membuat ETag adalah ketika Amazon S3 menambahkan tanda hubung dengan jumlah total bagian sampai akhir.

Misalnya, pertimbangkan objek yang diunggah dengan unggahan multibagian yang memiliki ETag `C9A5A6878D97B48CC965C1E41859F034-14`. Dalam hal ini, `C9A5A6878D97B48CC965C1E41859F034` adalah intisari MD5 dari semua intisari yang digabungkan bersama. `-14` menunjukkan bahwa ada 14 bagian yang terkait dengan unggahan multibagian objek ini.

Jika Anda telah mengaktifkan nilai checksum tambahan untuk objek multibagian Anda, Amazon S3 menghitung checksum untuk setiap bagian individual dengan menggunakan algoritma checksum yang ditentukan. Checksum untuk objek selesai dihitung dengan cara yang sama seperti Amazon S3 menghitung intisari MD5 untuk unggahan multibagian. Anda dapat menggunakan checksum ini untuk memverifikasi integritas objek.

Untuk mengambil informasi tentang objek, termasuk berapa banyak bagian yang membentuk seluruh objek, Anda dapat menggunakan GetObjectAttributesoperasi. Dengan checksum tambahan, Anda juga dapat memulihkan informasi untuk setiap bagian yang mencakup nilai checksum untuk setiap bagian.

Untuk upload selesai, Anda bisa mendapatkan checksum bagian individual dengan menggunakan HeadObjectoperasi GetObjectatau dan menentukan nomor bagian atau rentang byte yang sejajar dengan satu bagian. Jika Anda ingin mengambil nilai checksum untuk masing-masing bagian dari unggahan multibagian yang masih berlangsung, Anda dapat menggunakannya. ListParts

Oleh karena cara Amazon S3 menghitung checksum untuk objek multibagian, nilai checksum untuk objek mungkin berubah jika Anda menyalinnya. Jika Anda menggunakan SDK atau REST API dan

menelepon CopyObject, Amazon S3 menyalin objek apa pun hingga batasan ukuran `CopyObject` operasi API. Amazon S3 melakukan penyalinan ini sebagai tindakan tunggal, terlepas dari apakah objek diunggah dalam satu permintaan atau sebagai bagian dari unggahan multibagian. Dengan perintah salin, checksum objek adalah checksum langsung dari objek penuh. Jika awalnya objek tersebut diunggah menggunakan unggahan multibagian, maka nilai checksum berubah meskipun datanya belum berubah.

> **Note**
>
> Objek yang lebih besar dari batasan ukuran operasi API `CopyObject` harus menggunakan perintah salin multibagian.

> **Important**
>
> Saat Anda melakukan beberapa operasi menggunakan AWS Management Console, Amazon S3 menggunakan unggahan multibagian jika objek berukuran lebih besar dari 16 MB. Dalam hal ini, checksum itu bukan merupakan checksum langsung dari objek penuh, melainkan perhitungan berdasarkan nilai checksum dari masing-masing bagian.
> Misalnya, pertimbangkan objek berukuran 100 MB yang Anda unggah sebagai unggahan langsung satu bagian menggunakan API REST. Dalam hal ini, checksum adalah checksum dari seluruh objek. Jika nanti Anda menggunakan konsol untuk mengganti nama objek tersebut, menyalinnya, mengubah kelas penyimpanan, atau mengedit metadata, Amazon S3 menggunakan fungsionalitas unggahan multibagian untuk memperbarui objek. Akibatnya, Amazon S3 menciptakan nilai checksum baru untuk objek yang dihitung berdasarkan nilai checksum dari masing-masing bagian.
> Daftar operasi konsol sebelumnya bukanlah daftar lengkap dari semua kemungkinan tindakan yang dapat Anda lakukan sehingga Amazon S3 memperbarui objek menggunakan fungsionalitas unggahan multibagian. AWS Management Console Perlu diingat bahwa setiap kali Anda menggunakan konsol untuk bertindak pada objek yang berukuran lebih dari 16 MB, nilai checksum mungkin bukan checksum keseluruhan objek.

Menghapus objek Amazon S3

Anda dapat menghapus satu atau beberapa objek langsung dari Amazon S3 menggunakan konsol Amazon S3 AWS , SDK,AWS CLI(), atau AWS Command Line Interface REST API. Karena semua

objek dalam bucket S3 Anda mengeluarkan biaya penyimpanan, Anda sebaiknya menghapus objek yang tidak lagi dibutuhkan. Misalnya, jika Anda mengumpulkan file log, sebaiknya hapus file tersebut saat tidak diperlukan lagi. Anda dapat mengatur aturan siklus hidup untuk secara otomatis menghapus objek seperti file log. Untuk informasi selengkapnya, lihat the section called “Menetapkan konfigurasi siklus hidup”.

Untuk informasi tentang fitur dan harga Amazon S3, lihat Harga Amazon S3.

Anda memiliki opsi API berikut saat menghapus objek:

- Delete a single object–Amazon S3 menyediakan operasi API `DELETE` (`DeleteObject`) yang dapat Anda gunakan untuk menghapus satu objek dalam satu permintaan HTTP.
- Hapus beberapa objek–Amazon S3 menyediakan operasi API Multi-Object Delete (`DeleteObjects`) yang dapat Anda gunakan untuk menghapus hingga 1.000 objek dalam satu permintaan HTTP.

Saat menghapus objek dari bucket yang tidak mendukung versi, Anda hanya memberikan nama kunci objek. Namun, saat menghapus objek dari bucket yang mendukung versi, Anda dapat memberikan ID versi objek secara opsional untuk menghapus versi objek tertentu.

Secara terprogram menghapus objek dari sebuah bucket yang diaktifkan dengan versi

Jika bucket Anda diaktifkan dengan versi, maka beberapa versi dari objek yang sama dapat muncul dalam bucket tersebut. Saat menggunakan bucket yang diaktifkan dengan versi, operasi penghapusan API memungkinkan opsi berikut:

- Tentukan permintaan penghapusan non-versi–Menentukan hanya kunci objek, dan bukan ID versi. Dalam kasus ini, Amazon S3 menciptakan sebuah penanda hapus dan menampilkan ID versinya di dalam respons. Ini akan menghilangkan objek Anda dari bucket tersebut. Untuk informasi tentang pembuatan Penentuan Versi objek dan konsep penanda hapus, lihat Menggunakan Penentuan Versi dalam bucket S3.
- Tentukan permintaan penghapusan berversi–Menentukan kunci dan juga ID versi. Dalam hal ini, dua hasil berikut mungkin terjadi:
  - Jika ID versi memetakan ke versi objek tertentu, Amazon S3 menghapus versi objek tertentu.
  - Jika ID versi memetakan ke penanda penghapusan objek tersebut, Amazon S3 akan menghapus penanda penghapusan. Ini membuat objek tersebut muncul kembali di bucket Anda.

Menghapus objek dari bucket yang mengaktifkan MFA

Saat menghapus objek dari bucket yang mengaktifkan autentikasi multi-faktor (MFA), perhatikan hal berikut:

- Jika Anda menyediakan token MFA yang tidak valid, permintaan akan selalu gagal.
- Jika Anda memiliki bucket berkemampuan MFA dan Anda membuat permintaan penghapusan berversi (Anda memberikan kunci objek dan ID versi), permintaan tersebut akan gagal jika Anda tidak memberikan token MFA yang valid. Selain itu, saat menggunakan operasi API Penghapusan Multi-Objek pada bucket yang mendukung MFA, jika salah satu penghapusan merupakan permintaan penghapusan berversi (yaitu, Anda menentukan kunci objek dan ID versi), seluruh permintaan akan gagal jika Anda tidak memberikan token MFA.

Namun, dalam kasus berikut, permintaan berhasil:

- Jika Anda memiliki bucket berkemampuan MFA dan Anda membuat permintaan penghapusan yang tidak berversi (Anda tidak menghapus objek yang berversi), dan Anda tidak memberikan token MFA, maka penghapusannya berhasil.
- Jika Anda memiliki permintaan Penghapusan Multi-Objek yang hanya menentukan objek non-versi yang akan dihapus dari bucket yang mengaktifkan MFA dan Anda tidak memberikan token MFA, penghapusannya berhasil.

Untuk informasi tentang penghapusan MFA, lihat Mengonfigurasi penghapusan MFA.

Topik

- Menghapus satu objek
- Menghapus beberapa objek

Menghapus satu objek

Anda dapat menggunakan konsol Amazon S3 atau DELETE API untuk menghapus satu objek yang ada dari bucket S3. Untuk informasi selengkapnya tentang menghapus objek di Amazon S3, lihat Menghapus objek Amazon S3.

Karena semua objek dalam bucket S3 Anda mengeluarkan biaya penyimpanan, Anda sebaiknya menghapus objek yang tidak lagi dibutuhkan. Misalnya, jika Anda mengumpulkan file log, sebaiknya

hapus file tersebut saat tidak diperlukan lagi. Anda dapat mengatur aturan siklus hidup untuk secara otomatis menghapus objek seperti file log. Untuk informasi selengkapnya, lihat the section called "Menetapkan konfigurasi siklus hidup".

Untuk informasi tentang fitur dan harga Amazon S3, lihat Harga Amazon S3.

Menggunakan konsol S3

Ikuti langkah-langkah berikut ini untuk menggunakan konsol Amazon S3 guna menghapus satu objek dari bucket.

> **Warning**
>
> Saat Anda menghapus objek atau versi objek tertentu secara permanen di konsol Amazon S3, penghapusan tidak dapat dibatalkan.

Untuk menghapus objek yang memiliki versi diaktifkan atau ditangguhkan

> **Note**
>
> Jika ID versi untuk objek dalam bucket yang ditangguhkan versi ditandai sebagai`NULL`, S3 akan menghapus objek secara permanen karena tidak ada versi sebelumnya. Namun, jika ID versi yang valid dicantumkan untuk objek dalam bucket yang ditangguhkan versi, maka S3 membuat penanda hapus untuk objek yang dihapus, sambil mempertahankan versi objek sebelumnya.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Nama bucket, pilih nama bucket yang ingin Anda hapus objeknya.
3. Pilih objek dan kemudian pilih Hapus.
4. Untuk mengonfirmasi penghapusan daftar objek di bawah Objek yang ditentukan di objek Hapus? kotak teks, masukkan**`delete`**.

Untuk menghapus versi objek tertentu secara permanen dalam bucket berkemampuan versi

> **Warning**
>
> Saat Anda menghapus versi objek tertentu secara permanen di Amazon S3, penghapusan tidak dapat dibatalkan.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Nama bucket, pilih nama bucket yang ingin Anda hapus objeknya.
3. Pilih objek yang ingin Anda hapus.
4. Pilih sakelar Tampilkan versi.
5. Pilih versi objek dan kemudian pilih Hapus.
6. Untuk mengonfirmasi penghapusan permanen dari versi objek tertentu yang tercantum di bawah Objek yang ditentukan, di objek Hapus? kotak teks, masukkan Hapus secara permanen. Amazon S3 secara permanen menghapus versi objek tertentu.

Untuk menghapus objek secara permanen di bucket Amazon S3 yang tidak mengaktifkan versi

> **Warning**
>
> Saat Anda menghapus objek secara permanen di Amazon S3, penghapusan tidak dapat dibatalkan. Juga, untuk ember apa pun tanpa mengaktifkan versi, penghapusan bersifat permanen.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Nama bucket, pilih nama bucket yang ingin Anda hapus objeknya.
3. Pilih objek dan kemudian pilih Hapus.
4. Untuk mengonfirmasi penghapusan permanen objek yang terdaftar di bawah Objek yang ditentukan, di objek Hapus? kotak teks, masukkan hapus secara permanen.

> **Note**
>
> Jika Anda mengalami masalah dengan menghapus objek Anda, lihatSaya ingin menghapus objek berversi secara permanen.

Menggunakan AWS SDK

Contoh berikut menunjukkan bagaimana Anda dapat menggunakan AWS SDK untuk menghapus objek dari bucket. Untuk informasi selengkapnya, lihat DELETE Object dalam Referensi API Amazon Simple Storage Service

Jika Penentuan Versi S3 Anda telah diaktifkan pada bucket tersebut, Anda memiliki opsi berikut:

- Menghapus versi objek tertentu dengan menentukan ID versi.
- Menghapus objek tanpa menentukan ID versi, di mana Amazon S3 menambahkan sebuah penanda hapus ke objek.

Untuk informasi selengkapnya tentang Penentuan Versi S3, lihat Menggunakan Penentuan Versi dalam bucket S3.

Java

Example Contoh 1: Menghapus sebuah objek (bucket non-versi)

Contoh berikut mengasumsikan bahwa bucket tidak mengaktifkan Penentuan Versi, dan objek tidak memiliki ID versi apa pun. Dalam permintaan penghapusan, Anda hanya menentukan kunci objek dan bukan ID versi.

Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.DeleteObjectRequest;
```

```
import java.io.IOException;

public class DeleteObjectNonVersionedBucket {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ****";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            s3Client.deleteObject(new DeleteObjectRequest(bucketName, keyName));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example Contoh 2: Menghapus sebuah objek (bucket berversi)

Contoh berikut menampilkan penghapusan sebuah objek dari sebuah bucket berversi. Contoh ini menghapus versi objek tertentu dengan menentukan nama kunci objek dan ID versi.

Contoh ini melakukan hal berikut:

1. Menambahkan sebuah objek sampel ke bucket. Amazon S3 menampilkan ID versi dari objek yang baru saja ditambahkan. Contohnya menggunakan ID versi ini dalam permintaan penghapusan.
2. Hapus versi objek dengan menentukan nama kunci objek maupun ID versi. Jika tidak ada versi lain dari objek tersebut, Amazon S3 akan sepenuhnya menghapus objek tersebut. Jika tidak, Amazon S3 hanya menghapus versi yang ditentukan.

> **Note**
>
> Anda dapat memperoleh ID versi dari sebuah objek dengan mengirimkan permintaan `ListVersions`.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketVersioningConfiguration;
import com.amazonaws.services.s3.model.DeleteVersionRequest;
import com.amazonaws.services.s3.model.PutObjectResult;

import java.io.IOException;

public class DeleteObjectVersionEnabledBucket {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ****";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Check to ensure that the bucket is versioning-enabled.
            String bucketVersionStatus =
 s3Client.getBucketVersioningConfiguration(bucketName).getStatus();
            if (!bucketVersionStatus.equals(BucketVersioningConfiguration.ENABLED))
 {
                System.out.printf("Bucket %s is not versioning-enabled.",
 bucketName);
            } else {
                // Add an object.
```

```
                PutObjectResult putResult = s3Client.putObject(bucketName, keyName,
                        "Sample content for deletion example.");
                System.out.printf("Object %s added to bucket %s\n", keyName,
 bucketName);

                // Delete the version of the object that we just created.
                System.out.println("Deleting versioned object " + keyName);
                s3Client.deleteVersion(new DeleteVersionRequest(bucketName, keyName,
 putResult.getVersionId()));
                System.out.printf("Object %s, version %s deleted\n", keyName,
 putResult.getVersionId());
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Contoh berikut menunjukkan cara untuk menghapus sebuah objek dari bucket berversi maupun non-versi. Untuk informasi selengkapnya tentang Penentuan Versi S3, lihat Menggunakan Penentuan Versi dalam bucket S3.

Example Penghapusan sebuah objek dari sebuah bucket non-versi

Contoh berikut menampilkan penghapusan sebuah objek dari sebuah bucket non-versi. Contoh ini mengasumsikan bahwa objek tidak memiliki ID versi, sehingga Anda tidak menentukan ID versi. Anda hanya menentukan kunci objek.

Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
```

```
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class DeleteObjectNonVersionedBucketTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** object key ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            DeleteObjectNonVersionedBucketAsync().Wait();
        }
        private static async Task DeleteObjectNonVersionedBucketAsync()
        {
            try
            {
                var deleteObjectRequest = new DeleteObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName
                };

                Console.WriteLine("Deleting an object");
                await client.DeleteObjectAsync(deleteObjectRequest);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 deleting an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 deleting an object", e.Message);
            }
        }
    }
```

```
}
```

Example Penghapusan sebuah objek dari sebuah bucket berversi

Contoh C# berikut menampilkan penghapusan sebuah objek dari sebuah bucket berversi. Ini menghapus versi objek tertentu dengan menentukan nama kunci objek dan ID versi.

Kode ini melakukan tugas-tugas berikut:

1. Mengaktifkan S3 Penentuan Versi pada bucket yang Anda tentukan (jika S3 Penentuan Versi sudah diaktifkan, hal ini tidak berpengaruh).
2. Menambahkan sebuah objek sampel ke bucket. Amazon S3 merespons dengan menampilkan ID versi dari objek yang baru saja ditambahkan. Contohnya menggunakan ID versi ini dalam permintaan penghapusan.
3. Hapus versi objek dengan menentukan nama kunci objek maupun ID versi.

> **Note**
>
> Anda dapat memperoleh ID versi dari sebuah objek dengan mengirimkan permintaan `ListVersions`.
>
> ```
> var listResponse = client.ListVersions(new ListVersionsRequest { BucketName
> = bucketName, Prefix = keyName });
> ```

Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class DeleteObjectVersion
    {
        private const string bucketName = "*** versioning-enabled bucket name ***";
        private const string keyName = "*** Object Key Name ***";
```

```
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            CreateAndDeleteObjectVersionAsync().Wait();
        }

        private static async Task CreateAndDeleteObjectVersionAsync()
        {
            try
            {
                // Add a sample object.
                string versionID = await PutAnObject(keyName);

                // Delete the object by specifying an object key and a version ID.
                DeleteObjectRequest request = new DeleteObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    VersionId = versionID
                };
                Console.WriteLine("Deleting an object");
                await client.DeleteObjectAsync(request);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
deleting an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
deleting an object", e.Message);
            }
        }

        static async Task<string> PutAnObject(string objectKey)
        {
            PutObjectRequest request = new PutObjectRequest
            {
```

```
                BucketName = bucketName,
                Key = objectKey,
                ContentBody = "This is the content body!"
            };
            PutObjectResponse response = await client.PutObjectAsync(request);
            return response.VersionId;
        }
    }
}
```

PHP

Contoh ini menunjukkan cara menggunakan kelas dari versi 3 AWS SDK for PHP untuk menghapus objek dari bucket yang tidak berversi. Untuk informasi tentang penghapusan sebuah objek dari sebuah bucket berversi, lihat Penggunaan API REST.

Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

Contoh PHP berikut menghapus objek dari bucket. Karena contoh ini menunjukkan cara menghapus objek dari bucket yang non-versi, contoh ini hanya menyediakan nama bucket dan kunci objek (bukan ID versi) dalam permintaan penghapusan.

```
<?php

require 'vendor/autoload.php';

use Aws\S3\S3Client;
use Aws\S3\Exception\S3Exception;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// 1. Delete the object from the bucket.
try
{
    echo 'Attempting to delete ' . $keyname . '...' . PHP_EOL;
```

```
    $result = $s3->deleteObject([
        'Bucket' => $bucket,
        'Key'    => $keyname
    ]);

    if ($result['DeleteMarker'])
    {
        echo $keyname . ' was deleted or does not exist.' . PHP_EOL;
    } else {
        exit('Error: ' . $keyname . ' was not deleted.' . PHP_EOL);
    }
}
catch (S3Exception $e) {
    exit('Error: ' . $e->getAwsErrorMessage() . PHP_EOL);
}

// 2. Check to see if the object was deleted.
try
{
    echo 'Checking to see if ' . $keyname . ' still exists...' . PHP_EOL;

    $result = $s3->getObject([
        'Bucket' => $bucket,
        'Key'    => $keyname
    ]);

    echo 'Error: ' . $keyname . ' still exists.';
}
catch (S3Exception $e) {
    exit($e->getAwsErrorMessage());
}
```

Javascript

```
import { DeleteObjectCommand } from "@aws-sdk/client-s3";
import { s3Client } from "./libs/s3Client.js" // Helper function that creates Amazon
 S3 service client module.

export const bucketParams = { Bucket: "BUCKET_NAME", Key: "KEY" };

export const run = async () => {
  try {
    const data = await s3Client.send(new DeleteObjectCommand(bucketParams));
```

```
    console.log("Success. Object deleted.", data);
    return data; // For unit tests.
  } catch (err) {
    console.log("Error", err);
  }
};
run();
```

Menggunakan AWS CLI

Untuk menghapus satu objek per permintaan, gunakan API `DELETE`. Untuk informasi selengkapnya, lihat DELETE Object. Untuk informasi selengkapnya tentang menggunakan CLI untuk menghapus objek, lihat delete-object.

Penggunaan API REST

Anda dapat menggunakan AWS SDK untuk menghapus objek. Namun, jika aplikasi Anda memerlukannya, Anda dapat mengirimkan permintaan REST secara langsung. Untuk informasi selengkapnya, lihat DELETE Object dalam Referensi API Amazon Simple Storage Service.

Menghapus beberapa objek

Karena semua objek dalam bucket S3 Anda mengeluarkan biaya penyimpanan, Anda sebaiknya menghapus objek yang tidak lagi dibutuhkan. Misalnya, jika Anda mengumpulkan file log, sebaiknya hapus file tersebut saat tidak diperlukan lagi. Anda dapat mengatur aturan siklus hidup untuk secara otomatis menghapus objek seperti file log. Untuk informasi selengkapnya, lihat the section called “Menetapkan konfigurasi siklus hidup”.

Untuk informasi tentang fitur dan harga Amazon S3, lihat Harga Amazon S3.

Anda dapat menggunakan konsol Amazon S3, AWS SDK, atau REST API untuk menghapus beberapa objek secara bersamaan dari bucket S3.

Menggunakan konsol S3

Ikuti langkah-langkah berikut untuk menggunakan konsol Amazon S3 untuk menghapus beberapa objek dari bucket.

> **Warning**
>
> - Menghapus objek yang dipilih tadi tidak dapat diurungkan.
> - Tindakan ini menghapus semua objek yang telah ditentukan. Saat menghapus folder, tunggu hingga tindakan penghapusannya selesai sebelum menambahkan objek baru ke folder tersebut. Jika tidak, objek baru mungkin juga terhapus.
> - Saat menghapus objek dalam bucket tanpa mengaktifkan versi, Amazon S3 akan menghapus objek secara permanen.
> - Saat menghapus objek dalam bucket dengan versi bucket diaktifkan atau ditangguhkan, Amazon S3 membuat penanda hapus. Untuk informasi selengkapnya, lihat Bekerja dengan penanda hapus.

Untuk menghapus objek yang memiliki versi diaktifkan atau ditangguhkan

> **Note**
>
> Jika ID versi untuk objek dalam bucket yang ditangguhkan versi ditandai sebagai`NULL`, S3 akan menghapus objek secara permanen karena tidak ada versi sebelumnya. Namun, jika ID versi yang valid dicantumkan untuk objek dalam bucket yang ditangguhkan versi, maka S3 membuat penanda hapus untuk objek yang dihapus, sambil mempertahankan versi objek sebelumnya.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Dalam daftar nama Bucket, pilih nama bucket yang ingin Anda hapus objeknya.
3. Pilih objek dan kemudian pilih Hapus.
4. Untuk mengonfirmasi penghapusan daftar objek di bawah Objek yang ditentukan di objek Hapus? kotak teks, masukkan**`delete`**.

Untuk menghapus versi objek tertentu secara permanen dalam bucket berkemampuan versi

> **Warning**
>
> Saat Anda menghapus versi objek tertentu secara permanen di Amazon S3, penghapusan tidak dapat dibatalkan.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Dalam daftar nama Bucket, pilih nama bucket yang ingin Anda hapus objeknya.
3. Pilih objek yang ingin Anda hapus.
4. Pilih sakelar Tampilkan versi.
5. Pilih versi objek dan kemudian pilih Hapus.
6. Untuk mengonfirmasi penghapusan permanen dari versi objek tertentu yang tercantum di bawah Objek yang ditentukan, di objek Hapus? kotak teks, masukkan Hapus secara permanen. Amazon S3 secara permanen menghapus versi objek tertentu.

Untuk menghapus objek secara permanen di bucket Amazon S3 yang tidak mengaktifkan versi

> **Warning**
>
> Saat Anda menghapus objek secara permanen di Amazon S3, penghapusan tidak dapat dibatalkan. Juga, untuk ember apa pun tanpa mengaktifkan versi, penghapusan bersifat permanen.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Dalam daftar nama Bucket, pilih nama bucket yang ingin Anda hapus objeknya.
3. Pilih objek dan kemudian pilih Hapus.
4. Untuk mengonfirmasi penghapusan permanen objek yang terdaftar di bawah Objek yang ditentukan, di objek Hapus? kotak teks, masukkan hapus secara permanen.

> **Note**
>
> Jika Anda mengalami masalah dengan menghapus objek Anda, lihatSaya ingin menghapus objek berversi secara permanen.

Menggunakan AWS SDK

Untuk contoh cara menghapus beberapa objek dengan AWS SDK, lihatGunakan `DeleteObjects` dengan AWS SDK atau CLI.

Untuk informasi umum tentang penggunaan AWS SDK yang berbeda, lihatMengembangkan dengan Amazon S3 menggunakan SDK AWS.

Penggunaan API REST

Anda dapat menggunakan AWS SDK untuk menghapus beberapa objek menggunakan Multi-Object Delete API. Namun, jika aplikasi Anda memerlukannya, Anda dapat mengirimkan permintaan REST secara langsung.

Untuk informasi selengkapnya, lihat Menghapus Beberapa Objek dalam Referensi API Amazon Simple Storage Service.

Mengatur, membuat daftar, dan bekerja dengan objek Anda

Di Amazon S3, Anda dapat menggunakan prefiks untuk mengatur penyimpanan Anda. Prefiks adalah pembuatan grup logis dari objek dalam bucket. Nilai prefiks mirip dengan nama direktori yang memungkinkan Anda untuk menyimpan data serupa di bawah direktori yang sama dalam sebuah bucket. Ketika Anda mengunggah objek secara terprogram, Anda dapat menggunakan prefiks untuk organisasi data Anda.

Di konsol Amazon S3, prefiks disebut sebagai folder. Anda dapat melihat semua objek dan folder Anda di konsol S3 dengan menavigasi ke bucket. Anda juga dapat melihat informasi tentang setiap objek, termasuk properti objek.

Untuk informasi selengkapnya tentang daftar dan organisasi data Anda di Amazon S3, lihat topik berikut.

Topik

- Organisasi objek menggunakan prefiks
- Membuat daftar kunci objek secara terprogram
- Mengatur objek di konsol Amazon S3 dengan menggunakan folder
- Melihat gambaran umum objek di konsol Amazon S3
- Melihat properti objek di konsol Amazon S3

Organisasi objek menggunakan prefiks

Anda dapat menggunakan prefiks untuk organisasi data yang Anda simpan di bucket Amazon S3. Prefiks adalah string karakter di bagian awal nama kunci objek. Prefiks dapat memiliki panjang berapa pun, tergantung pada panjang maksimum dari nama kunci objek (1.024 byte). Anda dapat menganggap prefiks sebagai cara untuk mengatur data Anda dengan cara yang mirip dengan direktori. Namun, prefiks bukan direktori.

Pencarian berdasarkan prefiks membatasi hasil hanya pada kunci yang dimulai dengan prefiks tertentu. Pembatas menyebabkan operasi daftar menggabungkan semua kunci yang memiliki prefiks yang sama ke dalam satu hasil daftar ringkasan.

Tujuan dari parameter prefiks dan pembatas adalah untuk membantu Anda mengatur dan kemudian menelusuri kunci Anda secara hierarki. Untuk melakukan hal ini, pertama-tama pilih sebuah pembatas untuk bucket Anda, seperti garis miring (/), yang tidak ada dalam nama kunci mana pun yang Anda antisipasi. Anda dapat menggunakan karakter lain sebagai pembatas. Tidak ada yang unik tentang karakter garis miring (/), tetapi ini adalah pembatas prefiks yang sangat umum. Selanjutnya, buat nama kunci Anda dengan menggabungkan semua tingkat hierarki yang memuatnya, pisahkan setiap tingkat dengan pembatas.

Misalnya, jika Anda menyimpan informasi tentang kota, Anda mungkin mengaturnya berdasarkan benua, lalu berdasarkan negara, lalu berdasarkan provinsi atau negara bagian. Karena nama-nama ini biasanya tidak mengandung tanda baca, Anda dapat menggunakan garis miring (/) sebagai pembatas. Contoh berikut menggunakan pembatas garis miring (/).

- Eropa/Prancis/Nouvelle-Aquitaine/Bordeaux
- Amerika Utara/Kanada/Quebec/Montreal
- Amerika Utara/AS/Washington/Bellevue
- Amerika Utara/AS/Washington/Seattle

Jika Anda menyimpan data untuk setiap kota di dunia dengan cara ini, maka pengelolaan namespace kunci datar akan terasa aneh. Dengan menggunakan `Prefix` dan `Delimiter` dengan operasi daftar, Anda dapat menggunakan hierarki yang telah Anda buat untuk mendaftarkan data Anda. Misalnya, untuk membuat daftar semua negara bagian di AS, atur `Delimiter='/'` dan `Prefix='North America/USA/'`. Untuk membuat daftar semua provinsi di Kanada yang datanya Anda miliki, atur `Delimiter='/'` dan `Prefix='North America/Canada/'`.

Untuk informasi selengkapnya tentang pembatas, prefiks, dan folder bersarang, lihat Perbedaan antara folder prefiks dan bersarang.

Membuat daftar objek menggunakan prefiks dan pembatas

Jika Anda mengeluarkan permintaan daftar dengan pembatas, Anda dapat menelusuri hierarki hanya pada satu tingkat, melompati dan meringkas (mungkin jutaan) kunci yang ada di tingkat yang lebih dalam. Misalnya, asumsikan Anda memiliki bucket (*`DOC-EXAMPLE-BUCKET`*) dengan kunci-kunci berikut:

```
sample.jpg

photos/2006/January/sample.jpg

photos/2006/February/sample2.jpg

photos/2006/February/sample3.jpg

photos/2006/February/sample4.jpg
```

Bucket sampel hanya memiliki objek `sample.jpg` pada tingkat root. Untuk mencantumkan objek tingkat root dalam bucket saja, Anda dapat mengirimkan permintaan GET pada bucket dengan karakter pembatas garis miring (`/`). Sebagai respons, Amazon S3 mengembalikan kunci objek `sample.jpg` karena tidak mengandung karakter pembatas `/`. Semua kunci lainnya mengandung karakter pembatas. Amazon S3 mengelompokkan kunci ini dan menampilkan satu `CommonPrefixes` elemen dengan nilai prefiks `photos/`, yang merupakan sebuah substring dari awal kunci ini hingga peristiwa pertama dari pembatas yang ditentukan.

Example

```
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>DOC-EXAMPLE-BUCKET</Name>
```

```
  <Prefix></Prefix>
  <Marker></Marker>
  <MaxKeys>1000</MaxKeys>
  <Delimiter>/</Delimiter>
  <IsTruncated>false</IsTruncated>
  <Contents>
    <Key>sample.jpg</Key>
    <LastModified>2011-07-24T19:39:30.000Z</LastModified>
    <ETag>&quot;d1a7fb5eab1c16cb4f7cf341cf188c3d&quot;</ETag>
    <Size>6</Size>
    <Owner>
      <ID>75cc57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a</ID>
      <DisplayName>displayname</DisplayName>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
  <CommonPrefixes>
    <Prefix>photos/</Prefix>
  </CommonPrefixes>
</ListBucketResult>
```

Untuk informasi selengkapnya tentang membuat daftar kunci objek secara terprogram, lihat Membuat daftar kunci objek secara terprogram.

Membuat daftar kunci objek secara terprogram

Di Amazon S3, kunci dapat didaftarkan dengan prefiks. Anda dapat memilih prefiks umum untuk nama kunci terkait dan menandai kunci tersebut dengan karakter khusus yang membatasi hierarki. Anda kemudian dapat menggunakan operasi daftar untuk memilih dan menelusuri kunci secara hierarki. Hal ini mirip dengan bagaimana file disimpan dalam direktori dalam sistem file.

Amazon S3 mengekspos operasi daftar yang memungkinkan Anda menghitung kunci yang terdapat dalam bucket. Kunci yang dipilih untuk pendaftaran didasarkan pada bucket dan prefiks. Misalnya, bayangkan sebuah bucket bernama "`dictionary`" yang berisi kunci untuk setiap kata dalam bahasa Inggris. Anda dapat membuat panggilan untuk mencantumkan semua kunci dalam bucket yang dimulai dengan huruf "q". Hasil daftar selalu ditampilkan dalam urutan biner UTF-8.

Operasi daftar SOAP dan REST mengembalikan dokumen XML yang berisi nama kunci yang cocok dan informasi tentang objek yang diidentifikasi oleh setiap kunci.

> **Note**
>
> Dukungan SOAP melalui HTTP tidak digunakan lagi, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami sarankan Anda menggunakan REST API atau AWS SDK.

Kelompok kunci yang berbagi prefiks yang diakhiri oleh pembatas khusus dapat digabungkan dengan prefiks umum tersebut untuk tujuan pencatatan. Hal ini memungkinkan aplikasi untuk mengatur dan menelusuri kuncinya secara hierarki, seperti cara Anda mengatur file ke dalam direktori dalam sistem file.

Misalnya, untuk memperluas bucket kamus agar berisi lebih dari sekadar kata-kata bahasa Inggris, Anda dapat membentuk kunci dengan mengawali setiap kata dengan bahasanya dan pembatas, seperti "`French/logical`". Dengan menggunakan skema penamaan ini dan fitur daftar hierarki, Anda hanya dapat mengambil daftar kata-kata dalam bahasa Prancis. Anda juga dapat menelusuri daftar bahasa tingkat atas yang tersedia tanpa harus mengulangi semua kunci intervensi leksikografis. Untuk informasi selengkapnya tentang aspek pendaftaran ini, lihat Organisasi objek menggunakan prefiks.

API REST

Jika aplikasi Anda memerlukannya, Anda dapat mengirimkan permintaan REST secara langsung. Anda dapat mengirimkan permintaan GET untuk mengembalikan beberapa atau semua objek dalam bucket atau Anda dapat menggunakan kriteria pemilihan untuk mengembalikan subset objek dalam bucket. Untuk informasi selengkapnya, lihat GET Bucket (List Objects) Version 2 dalam Referensi API Amazon Simple Storage Service.

Membuat daftar implementasi efisiensi

Performa daftar tidak terpengaruh langsung oleh jumlah kunci dalam bucket Anda. Hal ini juga tidak terpengaruh dengan ada atau tidak adanya argumen `prefix`, `marker`, `maxkeys`, atau `delimiter`.

Iterasi melalui hasil multihalaman

Karena bucket dapat berisi kunci dalam jumlah yang hampir tidak terbatas, hasil lengkap dari kueri daftar bisa sangat besar. Untuk mengelola kumpulan hasil yang besar, API Amazon S3 mendukung penomoran halaman untuk membaginya menjadi beberapa respons. Setiap respons kunci daftar mengembalikan halaman hingga 1.000 kunci dengan indikator yang menunjukkan apakah respons

terpotong. Anda mengirim serangkaian permintaan kunci daftar sampai Anda menerima semua kunci. AWS Pustaka pembungkus SDK menyediakan pagination yang sama.

Contoh

Contoh kode berikut menunjukkan cara menggunakan`ListObjects`.

CLI

AWS CLI

Contoh berikut menggunakan `list-objects` perintah untuk menampilkan nama-nama semua objek dalam bucket yang ditentukan:

```
aws s3api list-objects --bucket text-content --query 'Contents[].{Key: Key, Size:
 Size}'
```

Contoh menggunakan `--query` argumen untuk memfilter output `list-objects` turun ke nilai kunci dan ukuran untuk setiap objek

Untuk informasi selengkapnya tentang objek, lihat Bekerja dengan Objek Amazon S3 di Panduan Pengembang Amazon S3.

- Untuk detail API, lihat ListObjectsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengambil informasi tentang semua item di bucket “test-files”.

```
Get-S3Object -BucketName test-files
```

Contoh 2: Perintah ini mengambil informasi tentang item "sample.txt" dari bucket “test-files”.

```
Get-S3Object -BucketName test-files -Key sample.txt
```

Contoh 3: Perintah ini mengambil informasi tentang semua item dengan awalan “sample” dari bucket “test-files”.

```
Get-S3Object -BucketName test-files -KeyPrefix sample
```

- Untuk detail API, lihat ListObjectsdi Referensi AWS Tools for PowerShell Cmdlet.

Mengatur objek di konsol Amazon S3 dengan menggunakan folder

Di Amazon S3, bucket dan objek adalah sumber daya utama, dan objek disimpan dalam bucket. Amazon S3 memiliki struktur datar, bukan hierarki seperti yang Anda lihat di sistem file. Namun, demi kemudahan organisasi, konsol Amazon S3 mendukung konsep folder sebagai pembuatan grup objek. Konsol melakukan ini dengan menggunakan prefiks nama bersama untuk objek yang dikelompokkan. Dengan kata lain, objek yang dikelompokkan memiliki nama yang dimulai dengan string umum. String umum ini, atau prefiks bersama, adalah nama folder. Nama objek juga disebut sebagai nama kunci.

Misalnya, Anda dapat membuat folder di konsol yang diberi nama `photos` dan menyimpan objek bernama `myphoto.jpg` di dalamnya. Objek kemudian disimpan dengan nama kunci `photos/myphoto.jpg`, tempat `photos/` adalah prefiksnya.

Berikut ini adalah dua contoh lainnya:

- Jika Anda memiliki tiga objek dalam bucket Anda—`logs/date1.txt`, `logs/date2.txt`, dan`logs/date3.txt`—konsol tersebut akan menampilkan folder bernama `logs`. Jika Anda membuka folder di konsol tersebut, anda akan melihat tiga objek: `date1.txt`, `date2.txt`, dan `date3.txt`.
- Jika Anda memiliki objek bernama `photos/2017/example.jpg`, konsol tersebut akan menunjukkan folder dengan nama `photos` yang berisi folder `2017`. Folder `2017` akan berisi objek `example.jpg`.

Anda dapat memiliki folder di dalam folder, tetapi bukan bucket di dalam bucket. Anda dapat mengunggah dan menyalin objek secara langsung ke dalam folder. Folder dapat dibuat, dihapus, dan dijadikan publik, tetapi folder tidak dapat diubah namanya. Objek dapat disalin dari satu folder ke folder lainnya.

> **Important**
>
> Saat Anda membuat folder di Amazon S3, S3 membuat objek 0 byte dengan kunci yang diatur ke nama folder yang Anda berikan. Misalnya, jika Anda membuat folder bernama

photos di dalam bucket, konsol Amazon S3 akan membuat objek 0 byte dengan kunci photos/ tersebut. Konsol membuat objek ini untuk mendukung gagasan folder. Konsol Amazon S3 memperlakukan semua objek yang memiliki karakter garis miring (/) sebagai karakter terakhir (trailing) dalam nama kunci sebagai folder (misalnya, examplekeyname/). Anda tidak dapat mengunggah objek yang memiliki nama kunci dengan karakter tambahan / dengan menggunakan konsol Amazon S3. Namun, Anda dapat mengunggah objek yang diberi nama dengan trailing / dengan Amazon S3 API dengan menggunakan AWS Command Line Interface AWS CLI() AWS , SDK, atau REST API. Objek yang diberi nama dengan trailing / muncul sebagai folder di konsol Amazon S3. Konsol Amazon S3 tidak menampilkan konten dan metadata untuk objek tersebut. Saat Anda menggunakan konsol untuk menyalin objek yang diberi nama dengan /, folder baru dibuat di lokasi tujuan, namun data dan metadata objek tidak disalin.

Topik

- Membuat folder
- Menjadikan folder publik
- Menghitung ukuran folder
- Menghapus folder

Membuat folder

Bagian ini menjelaskan cara menggunakan konsol Amazon S3 untuk membuat folder.

> **Important**
>
> Jika kebijakan bucket Anda mencegah pengunggahan objek ke bucket ini tanpa tag, metadata, atau penerima daftar kontrol akses (ACL), Anda tidak dapat membuat folder dengan menggunakan prosedur berikut. Sebaliknya, unggah folder kosong dan tentukan pengaturan berikut di konfigurasi unggahan.

Untuk membuat folder

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di panel navigasi kiri, pilih Bucket.
3. Di dalam daftar Bucket, pilih nama bucket yang ingin Anda buat foldernya.
4. Jika kebijakan bucket Anda mencegah pengunggahan objek ke bucket ini tanpa enkripsi, Anda harus memilih Aktifkan di bawah enkripsi di sisi server.
5. Pilih Buat folder.
6. Masukkan nama untuk folder (misalnya, **`favorite-pics`**). Kemudian, pilih Buat folder.

Menjadikan folder publik

Kami menyarankan untuk memblokir semua akses publik ke folder dan bucket Amazon S3 Anda, kecuali Anda secara khusus memerlukan folder atau bucket publik. Saat Anda membuat folder publik, siapa pun di internet dapat melihat semua objek yang dikelompokkan ke dalam folder itu.

Di konsol Amazon S3, Anda dapat membuat folder menjadi publik. Anda juga dapat menjadikan folder publik dengan membuat kebijakan bucket yang membatasi akses data berdasarkan prefiks. Untuk informasi selengkapnya, lihat Identity and Access Management untuk Amazon S3.

> **Warning**
>
> Setelah Anda menjadikan folder publik di konsol Amazon S3, Anda tidak dapat menjadikannya pribadi lagi. Sebaliknya, Anda harus mengatur izin untuk setiap objek individu di dalam folder publik sehingga objek tidak memiliki akses publik. Untuk informasi selengkapnya, lihat Mengonfigurasi ACL.

Topik

- Menghitung ukuran folder
- Menghapus folder

Menghitung ukuran folder

Bagian ini menjelaskan cara menggunakan konsol Amazon S3 untuk menghitung ukuran folder.

Untuk menghitung ukuran folder

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di panel navigasi kiri, pilih Bucket.
3. Di daftar Bucket, pilih nama bucket tempat folder Anda disimpan.
4. Pada daftar Objek, pilih kotak centang di samping nama folder.
5. Pilih Tindakan, lalu pilih Hitung ukuran total.

> **Note**
>
> Saat Anda keluar dari halaman, informasi folder (termasuk ukuran total) tidak akan tersedia lagi. Anda harus menghitung ukuran total lagi jika Anda ingin melihatnya lagi.

> **Important**
>
> - Saat Anda menggunakan tindakan Hitung ukuran total pada objek atau folder yang ditentukan dalam bucket, Amazon S3 menghitung jumlah total objek dan ukuran penyimpanan total. Namun, unggahan multibagian yang tidak lengkap atau sedang berlangsung dan versi sebelumnya atau yang tidak terkini tidak dihitung dalam jumlah total objek atau ukuran total. Tindakan ini hanya menghitung jumlah total objek dan ukuran total untuk versi saat ini atau terbaru dari setiap objek yang disimpan dalam bucket.
>
> Contoh, jika ada dua versi objek di bucket Anda, maka kalkulator penyimpanan di Amazon S3 menghitungnya hanya sebagai satu objek. Akibatnya, jumlah total objek yang dihitung di konsol Amazon S3 dapat berbeda dari metrik Hitungan Objek yang ditunjukkan di Lensa Penyimpanan S3 dan dari jumlah yang dilaporkan oleh metrik Amazon CloudWatch ,. `NumberOfObjects` Demikian juga, ukuran penyimpanan total juga dapat berbeda dari metrik Total Storage yang ditunjukkan pada S3 Storage Lens dan dari `BucketSizeBytes` metrik yang ditunjukkan pada CloudWatch.
> - Jika waktu untuk menghitung ukuran total folder besar terlalu lama, pertimbangkan untuk menggunakan Inventaris Amazon S3 dan Amazon S3 Select sebagai alternatif. Pertama, buat konfigurasi Inventaris S3 untuk menyertakan metadata Ukuran untuk setiap objek folder besar di laporan inventaris. Pengiriman laporan Inventaris S3 pertama dapat memakan waktu hingga 48 jam. Saat laporan inventaris diterbitkan, kueri laporan inventaris dengan `SUM` ekspresi S3 Select untuk menggabungkan ukuran objek dalam folder. Untuk informasi selengkapnya, lihat Mengonfigurasi inventaris dengan menggunakan konsol S3 dan Contoh SUM.

Menghapus folder

Bagian ini menjelaskan cara menggunakan konsol Amazon S3 untuk menghapus folder dari bucket S3.

Untuk informasi selengkapnya tentang fitur dan harga Amazon S3, lihat Amazon S3.

Untuk menghapus folder dari bucket S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Bucket, pilih nama bucket yang ingin Anda hapus foldernya.
3. Di daftar Objek, pilih kotak centang di sebelah folder dan objek yang ingin Anda hapus.
4. Pilih Hapus.
5. Di halaman Hapus objek, verifikasikan bahwa nama folder yang Anda pilih untuk penghapusan sudah terdaftar.
6. Di dalam kotak Hapus objek, masukkan **`delete`**, dan pilih Hapus objek.

> **Warning**
>
> Tindakan ini menghapus semua objek yang telah ditentukan. Saat menghapus folder, tunggu hingga tindakan penghapusannya selesai sebelum menambahkan objek baru ke folder tersebut. Jika tidak, objek baru mungkin juga terhapus.

Melihat gambaran umum objek di konsol Amazon S3

Anda dapat menggunakan konsol Amazon S3 untuk melihat gambaran umum sebuah objek. Konsol tersebut menyediakan semua informasi penting untuk sebuah objek di satu tempat.

Untuk membuka halaman detail untuk sebuah objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Bucket, pilih nama bucket yang berisi objek.
3. Di daftar Objek, pilih nama objek yang ingin Anda tinjau.

Halaman detail objek terbuka.

4. Untuk mengunduh objek, pilih Tindakan objek, lalu pilih Unduh. Untuk menyalin jalur objek ke clipboard, di bawah URL Objek, pilih URL tersebut.
5. Jika Penentuan Versi diaktifkan di bucket, pilih Versi untuk mendaftarkan versi objek.
   - Untuk mengunduh versi objek, pilih kotak centang di samping ID versi, pilih Tindakan, lalu pilih Unduh.
   - Untuk menghapus versi objek, pilih kotak centang di samping ID versi, dan pilih Hapus.

> **Important**
>
> Anda dapat membatalkan penghapusan objek hanya jika objek tersebut dihapus sebagai versi terbaru (saat ini). Anda tidak dapat membatalkan penghapusan objek versi sebelumnya.

Melihat properti objek di konsol Amazon S3

Anda dapat menggunakan konsol Amazon S3 untuk melihat properti sebuah objek, termasuk kelas penyimpanan, pengaturan enkripsi, tanda, dan metadata.

Untuk melihat properti sebuah objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Bucket, pilih nama bucket yang berisi objek.
3. Di dalam daftar Objek, pilih nama objek yang ingin Anda lihat propertinya.

   Ikhtisar objek untuk objek Anda terbuka. Anda dapat menggulir ke bawah untuk melihat properti objek.
4. Di halaman Ikhtisar objek, Anda dapat mengonfigurasi properti berikut untuk objek.

> **Note**
>
> - Jika Anda mengubah properti Kelas Penyimpanan, Enkripsi, atau Metadata, sebuah objek baru dibuat untuk menggantikan yang lama. Jika Penentuan Versi S3 diaktifkan,

versi baru objek akan dibuat, dan objek yang sudah ada menjadi versi yang lebih lama. Peran yang mengubah properti juga menjadi pemilik objek baru (atau versi objek).

- Jika Anda mengubah properti Storage Class, Encryption, atau Metadata untuk objek yang memiliki tag yang ditentukan pengguna, Anda harus memiliki izin. `s3:GetObjectTagging` Jika Anda mengubah properti ini untuk objek yang tidak memiliki tag yang ditentukan pengguna tetapi berukuran lebih dari 16 MB, Anda juga harus memiliki izin. `s3:GetObjectTagging`

  Jika kebijakan bucket tujuan menolak `s3:GetObjectTagging` tindakan, properti ini untuk objek akan diperbarui, tetapi tag yang ditentukan pengguna akan dihapus dari objek, dan Anda akan menerima kesalahan.

a. Kelas penyimpanan–Setiap objek di Amazon S3 memiliki kelas penyimpanan yang terkait dengannya. Kelas penyimpanan yang Anda pilih untuk digunakan tergantung pada seberapa sering Anda mengakses objeknya. Kelas penyimpanan default untuk objek S3 adalah STANDAR. Anda memilih kelas penyimpanan mana yang akan digunakan saat mengunggah sebuah objek. Untuk informasi selengkapnya tentang kelas penyimpanan, lihat Menggunakan kelas penyimpanan Amazon S3.

   Untuk mengubah kelas penyimpanan setelah Anda mengunggah objek, pilih Kelas penyimpanan. Pilih kelas penyimpanan yang Anda inginkan, lalu pilih Simpan.

b. Pengaturan enkripsi di sisi server–Anda dapat menggunakan enkripsi di sisi server untuk mengenkripsi objek S3 Anda. Untuk informasi selengkapnya, lihat Menentukan enkripsi di sisi server dengan AWS KMS (SSE-KMS) atau Menentukan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3).

c. Metadata–Setiap objek di Amazon S3 memiliki serangkaian pasangan nilai-nama yang mewakili metadata-nya. Untuk informasi tentang menambahkan metadata ke objek S3, lihat Mengedit metadata objek di konsol Amazon S3.

d. Tag–Anda mengkategorikan penyimpanan dengan menambahkan tanda untuk objek S3. Untuk informasi selengkapnya, lihat Mengategorikan penyimpanan Anda menggunakan tag.

e. Kunci objek penahanan dan retensi hukum — Anda dapat mencegah objek dihapus. Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek S3.

Bekerja dengan URL yang telah ditandatangani

Anda dapat menggunakan URL yang telah ditandatangani untuk memberikan akses terbatas waktu ke objek di Amazon S3 tanpa memperbarui kebijakan bucket Anda. URL yang telah ditandatangani dapat dimasukkan di dalam browser atau dapat digunakan oleh program untuk mengunduh objek. Kredensyal yang digunakan oleh URL presigned adalah milik AWS pengguna yang membuat URL.

Anda juga dapat menggunakan URL yang telah ditandatangani untuk mengizinkan seseorang mengunggah objek tertentu ke bucket Amazon S3 Anda. Ini memungkinkan unggahan tanpa mengharuskan pihak lain untuk memiliki kredensil atau izin AWS keamanan. Jika sebuah objek dengan kunci yang sama sudah ada di bucket seperti yang ditentukan dalam URL yang telah ditandatangani, Amazon S3 menggantikan objek yang sudah ada dengan objek yang diunggah.

Anda dapat menggunakan URL yang telah ditandatangani beberapa kali, hingga tanggal dan waktu kedaluwarsa.

Saat Anda membuat sebuah URL yang telah ditandatangani, Anda harus memberikan kredensial keamanan Anda, lalu menentukan berikut ini:

- Bucket Amazon S3
- Kunci objek (jika mengunduh objek ini akan ada di bucket Amazon S3 Anda, jika mengunggah ini adalah nama file yang akan diunggah)
- Metode HTTP (`GET` untuk mengunduh objek atau `PUT` untuk mengunggah)
- Interval waktu kedaluwarsa

Saat ini, URL yang telah ditandatangani Amazon S3 tidak mendukung penggunaan algoritma checksum integritas data berikut (CRC32, CRC32C, SHA-1, SHA-256) saat Anda mengunggah objek. Untuk memverifikasi integritas objek Anda setelah mengunggah, Anda dapat memberikan MD5 {i>digest<i} objek saat Anda mengunggahnya dengan URL yang telah ditandatangani. Untuk informasi selengkapnya tentang integritas objek, lihat Memeriksa integritas objek.

Topik

- Siapa yang dapat membuat URL yang telah ditandatangani
- Waktu kedaluwarsa untuk URL yang telah ditandatangani
- Pembatasan kemampuan URL yang telah ditandatangani
- Berbagi objek dengan URL yang telah ditandatangani

- Mengunggah objek dengan URL yang telah ditandatangani sebelumnya

Siapa yang dapat membuat URL yang telah ditandatangani

Siapa pun yang memiliki kredensial keamanan yang valid dapat membuat sebuah URL yang telah ditandatangani. Tetapi untuk seseorang berhasil mengakses objek, URL yang telah ditandatangani harus dibuat oleh seseorang yang memiliki izin untuk melakukan operasi yang menjadi dasar dari URL yang telah ditandatangani.

Berikut ini adalah jenis kredensial yang dapat Anda gunakan untuk membuat URL yang telah ditandatangani:

- Profil instans IAM–Valid hingga 6 jam.
- AWS Security Token Service–Berlaku hingga maksimum 36 jam saat ditandatangani dengan kredensial keamanan jangka panjang atau durasi kredensial sementara, mana yang berakhir terlebih dulu.
- Pengguna IAM - Berlaku hingga 7 hari saat Anda menggunakan AWS Signature Version 4.

  Untuk membuat sebuah URL yang telah ditandatangani yang valid hingga 7 hari, pertama-tama delegasikan kredensial pengguna IAM (kunci akses dan kunci rahasia) ke metode yang Anda gunakan untuk membuat URL yang telah ditandatangani.

> **Note**
>
> Jika Anda membuat URL yang telah ditandatangani menggunakan kredensial sementara, URL tersebut akan kedaluwarsa saat kredensial-nya kedaluwarsa. Secara umum, URL yang telah ditetapkan sebelumnya akan kedaluwarsa saat kredensi yang Anda gunakan untuk membuatnya dicabut, dihapus, atau dinonaktifkan. Ini benar terjadi meskipun jika URL dibuat dengan waktu kedaluwarsa yang lebih lama. Untuk masa pakai kredensil keamanan sementara, lihat Membandingkan operasi AWS STS API di Panduan Pengguna IAM.

Waktu kedaluwarsa untuk URL yang telah ditandatangani

URL yang telah ditandatangani tetap valid untuk jangka waktu yang ditentukan saat URL dibuat. Jika Anda membuat URL yang telah ditandatangani dengan konsol Amazon S3, waktu kedaluwarsa

dapat diatur antara 1 menit dan 12 jam. Jika Anda menggunakan AWS CLI atau AWS SDK, waktu kedaluwarsa dapat diatur setinggi 7 hari.

Jika Anda membuat URL yang telah ditetapkan sebelumnya dengan menggunakan token sementara, URL akan kedaluwarsa saat token kedaluwarsa. Secara umum, URL yang telah ditetapkan sebelumnya akan kedaluwarsa saat kredensi yang Anda gunakan untuk membuatnya dicabut, dihapus, atau dinonaktifkan. Ini benar terjadi meskipun jika URL dibuat dengan waktu kedaluwarsa yang lebih lama. Untuk informasi selengkapnya tentang bagaimana kredensial yang Anda gunakan memengaruhi waktu kedaluwarsanya, lihat Siapa yang dapat membuat URL yang telah ditandatangani.

Amazon S3 memeriksa tanggal dan waktu kedaluwarsa URL yang ditandatangani pada saat permintaan HTTP. Misalnya, jika klien mulai mengunduh file besar segera sebelum waktu kedaluwarsa, pengunduhan berlanjut meskipun waktu kedaluwarsa berlalu selama pengunduhan. Akan tetapi, jika koneksi menurun dan klien mencoba memulai ulang unduhan setelah waktu kedaluwarsa berlalu, pengunduhan gagal.

Pembatasan kemampuan URL yang telah ditandatangani

Kemampuan URL yang telah ditandatangani dibatasi oleh izin pengguna yang membuatnya. Pada intinya, URL yang telah ditandatangani adalah token pembawa yang memberikan akses kepada mereka yang memilikinya. Oleh karena itu, kami menyarankan agar Anda melindunginya sebagaimana mestinya. Berikut ini adalah beberapa metode yang dapat Anda gunakan untuk membatasi penggunaan URL yang telah ditandatangani Anda.

AWS Tanda Tangan Versi 4 (SiGv4)

Untuk menerapkan perilaku tertentu saat permintaan URL yang telah ditandatangani diautentikasi dengan menggunakan AWS Signature Version 4 (SigV4), Anda dapat menggunakan kunci kondisi dalam kebijakan bucket dan kebijakan titik akses. Contoh, kebijakan bucket berikut menggunakan `s3:signatureAge` kondisi untuk menolak permintaan URL yang telah ditandatangani Amazon S3 pada objek di *`example-s3-bucket1`* bucket jika tanda tangannya sudah lebih dari 10 menit. Untuk menggunakan contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Deny a presigned URL request if the signature is more than 10 min
 old",
```

```
            "Effect": "Deny",
            "Principal": {"AWS":"*"},
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::example-s3-bucket1/*",
            "Condition": {
               "NumericGreaterThan": {
                  "s3:signatureAge": 600000
               }
            }
         }
      ]
}
```

Untuk informasi selengkapnya tentang kunci kebijakan terkait AWS Tanda Tangan Versi 4, lihat Otentikasi AWS Tanda Tangan Versi 4 di Referensi API Amazon Simple Storage Service.

Pembatasan jalur jaringan

Jika Anda ingin membatasi penggunaan URL yang telah ditetapkan sebelumnya dan semua akses Amazon S3 ke jalur jaringan tertentu, Anda dapat menulis AWS Identity and Access Management kebijakan (IAM). Anda dapat mengatur kebijakan ini pada pengguna utama IAM yang melakukan panggilan, bucket Amazon S3, atau keduanya.

Pembatasan jalur jaringan pada pengguna utama IAM mengharuskan pengguna kredensial tersebut untuk membuat permintaan dari jaringan yang ditentukan. Pembatasan pada bucket atau titik akses mengharuskan semua permintaan ke sumber daya tersebut berasal dari jaringan tertentu. Pembatasan ini juga berlaku di luar skenario URL yang telah ditandatangani sebelumnya.

Kunci kondisi global IAM yang Anda gunakan bergantung pada jenis titik akhir. Jika Anda menggunakan titik akhir publik untuk Amazon S3, gunakan `aws:SourceIp`. Jika Anda menggunakan titik akhir cloud privat virtual (VPC) ke Amazon S3, gunakan `aws:SourceVpc` atau `aws:SourceVpce`.

Pernyataan kebijakan IAM berikut mengharuskan prinsipal untuk mengakses AWS hanya dari rentang jaringan yang ditentukan. Dengan pernyataan kebijakan ini, semua akses harus berasal dari rentang itu. Kebijakan ini juga termasuk kasus seseorang yang menggunakan URL yang telah ditandatangani sebelumnya untuk Amazon S3. Untuk menggunakan contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
{
    "Sid": "NetworkRestrictionForIAMPrincipal",
```

```
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
        "NotIpAddressIfExists": {"aws:SourceIp": "IP-address-range"},
        "BoolIfExists": {"aws:ViaAWSService": "false"}
    }
}
```

Untuk contoh kebijakan bucket tambahan yang menggunakan kunci kondisi `aws:SourceIp` AWS global untuk membatasi akses ke bucket Amazon S3 ke rentang jaringan tertentu, lihat. Mengelola akses berdasarkan alamat IP tertentu

Berbagi objek dengan URL yang telah ditandatangani

Semua objek Amazon S3 secara default bersifat privat, hanya pemilik objek yang memiliki izin untuk mengaksesnya. Namun, pemilik objek dapat berbagi objek dengan orang lain dengan membuat URL yang telah ditandatangani sebelumnya. URL yang telah ditetapkan sebelumnya menggunakan kredensial keamanan untuk memberikan izin terbatas waktu untuk mengunduh objek. URL dapat dimasukkan di dalam browser, atau digunakan oleh program untuk mengunduh objek. Kredensyal yang digunakan oleh URL presigned adalah milik AWS pengguna yang membuat URL.

Untuk informasi umum tentang URL yang telah ditandatangani, lihat. Bekerja dengan URL yang telah ditandatangani

Anda dapat membuat URL yang telah ditetapkan sebelumnya untuk berbagi objek tanpa menulis kode apa pun menggunakan konsol Amazon S3 AWS , Explorer for Visual Studio (Windows), atau. AWS Toolkit for Visual Studio Code Anda juga dapat membuat URL presigned secara terprogram menggunakan AWS Command Line Interface (AWS CLI) atau SDK. AWS

Menggunakan konsol S3

Anda dapat menggunakan konsol Amazon S3 untuk menghasilkan URL yang telah ditandatangani untuk berbagi objek dengan mengikuti langkah-langkah berikut. Saat menggunakan konsol, waktu kedaluwarsa maksimum untuk URL yang telah ditandatangani sebelumnya adalah 12 jam dari waktu pembuatan.

Untuk menghasilkan URL yang telah ditandatangani sebelumnya menggunakan konsol Amazon S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di panel navigasi kiri, pilih Bucket.
3. Di dalam daftar Bucket, pilih nama bucket yang berisi objek yang Anda inginkan sebagai URL yang telah ditandatangani.
4. Dalam daftar Objek, pilih objek yang ingin Anda buatkan URL yang telah ditandatangani.
5. Pada menu Tindakan objek, pilih Bagikan dengan URL yang telah ditandatangani.
6. Tentukan berapa lama Anda menginginkan URL yang telah ditandatangani tersebut valid.
7. Pilih Buat URL yang telah ditandatangani.
8. Ketika konfirmasi muncul, URL secara otomatis disalin ke clipboard Anda. Anda akan melihat tombol untuk menyalin URL yang telah ditandatangani jika Anda perlu menyalinnya lagi.

Menggunakan AWS CLI

AWS CLI Perintah contoh berikut menghasilkan URL yang telah ditetapkan sebelumnya untuk berbagi objek dari bucket Amazon S3. Saat Anda menggunakan AWS CLI, waktu kedaluwarsa maksimum untuk URL yang telah ditetapkan sebelumnya adalah 7 hari dari waktu pembuatan. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3 presign s3://example-s3-bucket1/mydoc.txt --expires-in 604800
```

> **Note**
>
> Untuk semua yang Wilayah AWS diluncurkan setelah 20 Maret 2019 Anda perlu menentukan `endpoint-url` dan `Wilayah AWS` dengan permintaan. Untuk daftar semua Wilayah dan titik akhir Amazon S3, lihat Wilayah dan Titik Akhir dalam AWS Referensi Umum.

```
aws s3 presign s3://example-s3-bucket1/mydoc.txt --expires-in 604800 --region af-
south-1 --endpoint-url https://s3.af-south-1.amazonaws.com
```

Untuk informasi selengkapnya, lihat presign dalam AWS CLI Referensi Perintah.

Menggunakan AWS SDK

Untuk contoh menggunakan AWS SDK untuk menghasilkan URL yang telah ditetapkan sebelumnya untuk berbagi objek, lihat Membuat URL presigned untuk Amazon S3 menggunakan SDK. AWS

Saat Anda menggunakan AWS SDK untuk menghasilkan URL yang telah ditetapkan sebelumnya, waktu kedaluwarsa maksimum adalah 7 hari dari waktu pembuatan.

> **Note**
>
> Untuk semua yang Wilayah AWS diluncurkan setelah 20 Maret 2019 Anda perlu menentukan `endpoint-url` dan `Wilayah AWS` dengan permintaan. Untuk daftar semua Wilayah dan titik akhir Amazon S3, lihat Wilayah dan Titik Akhir dalam AWS Referensi Umum.

> **Note**
>
> Saat menggunakan AWS SDK, atribut Tagging harus berupa header dan bukan parameter kueri. Semua atribut lainnya dapat diteruskan sebagai parameter untuk URL presigned.

Menggunakan AWS Toolkit for Visual Studio (Windows)

> **Note**
>
> Saat ini, AWS Toolkit for Visual Studio tidak mendukung Visual Studio untuk Mac.

1. Instal AWS Toolkit for Visual Studio menggunakan petunjuk berikut, Menginstal dan menyiapkan Toolkit for Visual Studio di AWS Toolkit for Visual Studio Panduan Pengguna.
2. Connect untuk AWS menggunakan langkah-langkah berikut, Connecting to AWS di Panduan AWS Toolkit for Visual Studio Pengguna.
3. Di panel sisi kiri berlabel AWS Explorer, klik dua kali bucket yang berisi objek Anda.
4. Klik kanan objek yang ingin Anda buat URL yang telah ditetapkan sebelumnya dan pilih Buat URL Pra-Tanda Tangan... .
5. Di jendela pop-up, atur tanggal kedaluwarsa dan waktu untuk URL yang telah ditetapkan sebelumnya.
6. Object Key, harus diisi terlebih dahulu berdasarkan objek yang Anda pilih.
7. Pilih GET untuk menentukan bahwa URL yang telah ditandatangani sebelumnya ini akan digunakan untuk mengunduh objek.

8. Pilih tombol Hasilkan.
9. Untuk menyalin URL ke clipboard, pilih Salin.
10. Untuk menggunakan URL presigned yang dihasilkan, tempel URL ke browser apa pun.

Menggunakan AWS Toolkit for Visual Studio Code

Jika Anda menggunakan Visual Studio Code, Anda dapat membuat URL yang telah ditandatangani sebelumnya untuk membagikan objek tanpa menulis kode apa pun dengan menggunakan AWS Toolkit for Visual Studio Code. Untuk informasi umum, lihat AWS Toolkit for Visual Studio Code dalam Panduan Pengguna AWS Toolkit for Visual Studio Code .

Untuk petunjuk tentang cara menginstal AWS Toolkit for Visual Studio Code, lihat Menginstal AWS Toolkit for Visual Studio Code di Panduan AWS Toolkit for Visual Studio Code Pengguna.

1. Connect untuk AWS menggunakan langkah-langkah berikut, Connecting to AWS Toolkit for Visual Studio Code di Panduan AWS Toolkit for Visual Studio Code Pengguna.
2. Pilih AWS logo di panel kiri di Visual Studio Code.
3. Di bawah EXPLORER, pilih S3.
4. Pilih bucket dan filenya, lalu buka menu konteks (klik kanan).
5. Pilih Hasilkan URL yang telah ditandatangani sebelumnya, lalu atur waktu kedaluwarsa (dalam hitungan menit).
6. Tekan Enter, dan URL yang telah ditandatangani sebelumnya akan disalin ke clipboard Anda.

Mengunggah objek dengan URL yang telah ditandatangani sebelumnya

Anda dapat menggunakan URL yang telah ditandatangani sebelumnya untuk mengizinkan seseorang mengunggah objek ke bucket Amazon S3 Anda. Menggunakan URL yang telah ditetapkan sebelumnya akan memungkinkan unggahan tanpa mengharuskan pihak lain memiliki kredensyal atau izin AWS keamanan. URL yang telah ditandatangani sebelumnya dibatasi oleh izin pengguna yang membuatnya. Artinya, jika Anda menerima URL yang telah ditandatangani sebelumnya untuk mengunggah suatu objek, Anda dapat mengunggah objek hanya jika pembuat URL tersebut memiliki izin yang diperlukan untuk mengunggah objek tersebut.

Saat seseorang menggunakan URL untuk mengunggah sebuah objek, Amazon S3 membuat objek dalam bucket yang ditentukan. Jika objek dengan kunci yang sama yang ditentukan dalam URL yang

telah ditandatangani sebelumnya sudah ada di bucket, Amazon S3 akan menggantikan objek yang ada dengan objek yang diunggah. Setelah diunggah, pemilik bucket akan memiliki objek tersebut.

Untuk informasi umum tentang URL yang telah ditandatangani, lihat. Bekerja dengan URL yang telah ditandatangani

Anda dapat membuat URL yang telah ditandatangani sebelumnya untuk mengunggah objek tanpa menulis kode apa pun dengan menggunakan AWS Explorer for Visual Studio. Anda juga dapat membuat URL yang telah ditandatangani sebelumnya secara terprogram menggunakan SDK AWS .

Menggunakan AWS Toolkit for Visual Studio (Windows)

> **Note**
>
> Saat ini, AWS Toolkit for Visual Studio tidak mendukung Visual Studio untuk Mac.

1. Instal AWS Toolkit for Visual Studio menggunakan petunjuk berikut, Menginstal dan menyiapkan Toolkit for Visual Studio di AWS Toolkit for Visual Studio Panduan Pengguna.
2. Connect untuk AWS menggunakan langkah-langkah berikut, Connecting to AWS di Panduan AWS Toolkit for Visual Studio Pengguna.
3. Di panel sisi kiri berlabel AWS Explorer, klik kanan bucket yang ingin Anda unggah objek.
4. Pilih Buat URL Pra-Tanda Tandatangani... .
5. Di jendela pop-up, atur tanggal kedaluwarsa dan waktu untuk URL yang telah ditetapkan sebelumnya.
6. Untuk Object Key, atur nama file yang akan diunggah. File yang Anda unggah harus sama persis dengan nama ini. Jika objek dengan kunci objek yang sama sudah ada di bucket, Amazon S3 akan mengganti objek yang ada dengan objek yang baru diunggah.
7. Pilih PUT untuk menentukan bahwa URL yang telah ditandatangani sebelumnya ini akan digunakan untuk mengunggah objek.
8. Pilih tombol Hasilkan.
9. Untuk menyalin URL ke clipboard, pilih Salin.
10. Untuk menggunakan URL ini, Anda dapat mengirim permintaan PUT dengan perintah `curl`. Sertakan path lengkap ke file Anda dan URL presigned itu sendiri.

```
curl -X PUT -T "/path/to/file" "presigned URL"
```

Menggunakan AWS SDK

Untuk contoh menggunakan AWS SDK untuk menghasilkan URL yang telah ditetapkan sebelumnya untuk mengunggah objek, lihat Membuat URL yang telah ditetapkan sebelumnya untuk Amazon S3 menggunakan SDK. AWS

Saat Anda menggunakan AWS SDK untuk menghasilkan URL yang telah ditetapkan sebelumnya, waktu kedaluwarsa maksimum adalah 7 hari dari waktu pembuatan.

> **Note**
>
> Untuk semua yang Wilayah AWS diluncurkan setelah 20 Maret 2019 Anda perlu menentukan `endpoint-url` dan `Wilayah AWS` dengan permintaan. Untuk daftar semua Wilayah dan titik akhir Amazon S3, lihat Wilayah dan Titik Akhir dalam AWS Referensi Umum.

Mengubah objek dengan S3 Lambda Objek

Dengan Lambda Objek Amazon S3, Anda dapat menambahkan kode Anda sendiri ke `GET`, `LIST`, dan `HEAD` Amazon S3 dan meminta untuk mengubah dan memproses data saat dikembalikan ke aplikasi. Anda dapat menggunakan kode khusus untuk memodifikasi data yang dikembalikan oleh permintaan `GET` S3 untuk melakukan filter baris, mengubah ukuran dan watermark gambar secara dinamis, menyamarkan data rahasia, dan banyak lagi. Anda juga dapat menggunakan S3 Lambda Objek untuk memodifikasi output permintaan `LIST` S3 untuk membuat tampilan kustom dari semua objek dalam bucket dan permintaan `HEAD` S3 untuk memodifikasi metadata objek, seperti nama objek dan ukuran. Anda dapat menggunakan S3 Object Lambda sebagai asal distribusi CloudFront Amazon Anda untuk menyesuaikan data bagi pengguna akhir, seperti mengubah ukuran gambar secara otomatis, mentranskode format lama (seperti dari JPEG ke WebP), atau menghapus metadata. Untuk informasi selengkapnya, lihat posting AWS Blog Menggunakan Amazon S3 Object Lambda dengan Amazon. CloudFront Didukung oleh fungsi AWS Lambda, kode Anda berjalan pada infrastruktur yang sepenuhnya dikelola oleh. AWS Menggunakan S3 Lambda Objek mengurangi kebutuhan untuk membuat dan menyimpan salinan turunan data Anda atau menjalankan proksi, semuanya tanpa perlu mengubah aplikasi Anda.

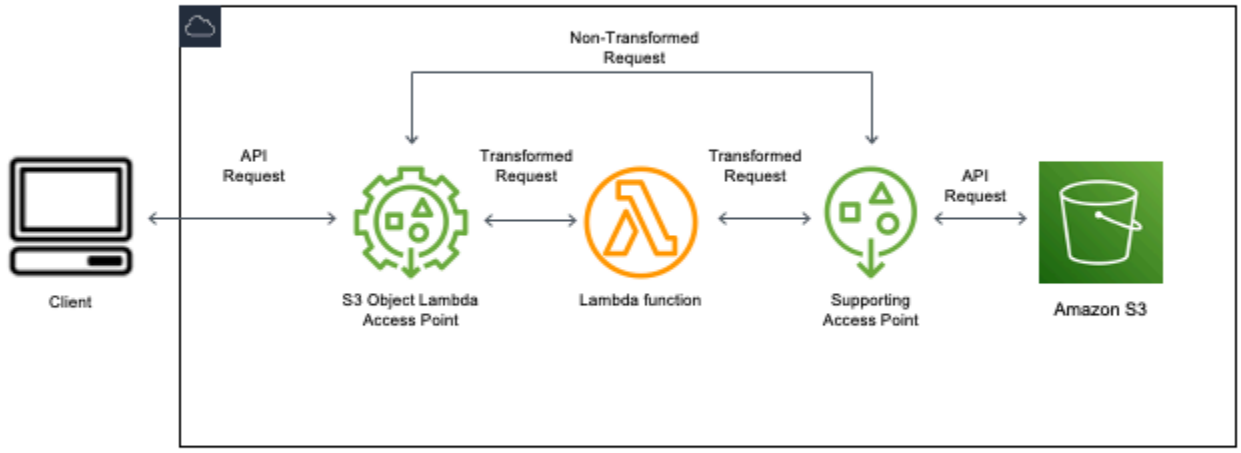
Bagaimana S3 Lambda Objek bekerja

S3 Object Lambda AWS Lambda menggunakan fungsi untuk secara otomatis memproses output standar `GET` S3`LIST`,, atau permintaan. `HEAD` AWS Lambda adalah layanan komputasi tanpa server

yang menjalankan kode yang ditentukan pelanggan tanpa memerlukan pengelolaan sumber daya komputasi yang mendasarinya. Anda dapat melakukan otorisasi dan menjalankan fungsi Lambda kustom Anda sendiri, menyesuaikan transformasi data untuk kasus penggunaan tertentu Anda.

Setelah Anda mengonfigurasi fungsi Lambda, Anda melampirkannya ke S3 Lambda Objek titik akhir layanan, yang dikenal sebagai Titik Akses Objek Lambda. Titik Akses Objek Lambda menggunakan titik akses S3 standar, yang dikenal sebagai titik akses pendukung, untuk mengakses Amazon S3.

Saat Anda mengirim permintaan ke Titik Akses Lambda Objek, Amazon S3 secara otomatis memanggil fungsi Lambda Anda. Data yang diambil dengan menggunakan permintaan S3 `GET`, `LIST`, atau `HEAD` melalui Titik Akses Lambda Objek mengembalikan hasil berubah kembali ke aplikasi. Semua permintaan lainnya diproses seperti biasa, seperti yang digambarkan dalam diagram berikut.



Topik di bagian ini menjelaskan cara bekerja dengan S3 Lambda Objek.

Topik

- Membuat Titik Akses Objek Lambda
- Menggunakan Titik Akses Lambda Objek Amazon S3
- Pertimbangan keamanan untuk Titik Akses S3 Lambda Objek

- Menulis fungsi Lambda untuk Titik Akses S3 Lambda Objek
- Menggunakan fungsi Lambda yang AWS dibangun
- Praktik terbaik dan pedoman untuk S3 Lambda Objek
- Tutorial S3 Lambda Objek
- Men-debug S3 Lambda Objek

Membuat Titik Akses Objek Lambda

Titik Akses Lambda Objek dikaitkan dengan tepat satu titik akses standar dan dengan demikian satu bucket Amazon S3. Untuk membuat titik akses Lambda Objek, Anda memerlukan sumber daya berikut:

- Bucket Amazon S3. Untuk informasi tentang membuat bucket, lihat the section called “Membuat bucket”.
- Titik akses standar S3. Saat Anda bekerja dengan Titik Akses Lambda Objek, titik akses standar ini dikenal sebagai titik akses pendukung. Untuk informasi selengkapnya tentang cara membuat titik akses standar, lihat the section called “Membuat titik akses”.
- Sebuah AWS Lambda fungsi. Anda dapat membuat fungsi Lambda Anda sendiri, atau Anda juga dapat menggunakan fungsi bawaan. Untuk informasi selengkapnya tentang membuat fungsi Lambda, lihat the section called “Menulis fungsi Lambda”. Untuk informasi lebih lanjut tentang fungsi bawaan, lihat Menggunakan fungsi Lambda yang AWS dibangun.
- (Opsional) Kebijakan AWS Identity and Access Management (IAM). Titik akses Amazon S3 mendukung kebijakan sumber daya IAM yang dapat Anda gunakan untuk mengontrol penggunaan titik akses berdasarkan sumber daya, pengguna, atau ketentuan lainnya. Untuk informasi selengkapnya tentang pembuatan kebijakan ini, lihat the section called “Mengonfigurasi kebijakan IAM”.

Bagian berikut menjelaskan cara membuat Titik Akses Lambda Objek dengan menggunakan:

- The AWS Management Console
- AWS Command Line Interface (AWS CLI)
- AWS CloudFormation Template
- The AWS Cloud Development Kit (AWS CDK)

Untuk informasi lebih lanjut tentang cara membuat Titik Akses Lambda Objek menggunakan API REST, lihat `CreateAccessPointForObjectLambda` dalam Referensi API Amazon Simple Storage Service.

Membuat Titik Akses Lambda Object

Gunakan salah satu dari prosedur berikut ini untuk membuat Titik Akses Lambda Object Anda.

Menggunakan konsol S3

Untuk membuat Titik Akses Lambda Objek menggunakan konsol

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di bilah navigasi, pilih nama yang saat ini ditampilkan Wilayah AWS. Selanjutnya, pilih Wilayah yang ingin Anda alihkan.
3. Di panel navigasi sebelah kiri, pilih Titik Akses Lambda Objek.
4. Pada halaman Titik Akses Lambda Objek, pilih Buat Titik Akses Lambda Objek.
5. Untuk Nama Titik Akses Lambda Objek, masukkan nama yang ingin Anda gunakan untuk titik aksesnya.

   Seperti halnya titik akses standar, ada aturan untuk penamaan Titik Akses Lambda Objek. Untuk informasi selengkapnya, lihat Aturan penamaan titik akses Amazon S3.
6. Untuk Titik Akses Dukungan, masukkan atau telusuri titik akses standar yang ingin Anda gunakan. Titik akses harus Wilayah AWS sama dengan objek yang ingin Anda ubah. Untuk informasi selengkapnya tentang cara membuat titik akses standar, lihat the section called “Membuat titik akses”.
7. Di bawah Konfigurasi transformasi, Anda dapat menambahkan fungsi yang mengubah data untuk Titik Akses Lambda Objek Anda. Lakukan salah satu hal berikut ini:

   - Jika Anda sudah memiliki AWS Lambda fungsi di akun Anda, Anda dapat memilihnya di bawah fungsi Invoke Lambda. Di sini Anda dapat memasukkan Nama Sumber Daya Amazon (ARN) dari fungsi Lambda di Anda atau Akun AWS memilih fungsi Lambda dari menu tarik-turun.
   - Jika Anda ingin menggunakan fungsi yang AWS dibangun pilih nama fungsi di bawah fungsi yang AWS dibangun dan pilih Buat fungsi Lambda. Ini akan membawa Anda ke konsol Lambda di mana Anda dapat menerapkan fungsi bawaan ke dalam konsol Anda. Akun AWS

Untuk informasi lebih lanjut tentang fungsi bawaan, lihat Menggunakan fungsi Lambda yang AWS dibangun.

Di bawah API S3, pilih satu atau beberapa operasi API untuk dipanggil. Untuk setiap API yang dipilih, Anda harus menentukan fungsi Lambda yang akan dipanggil.

8. (Opsional) Di bawah Muatan, tambahkan teks JSON yang ingin Anda berikan ke fungsi Lambda Anda sebagai input. Anda dapat mengonfigurasi muatan dengan parameter berbeda untuk Titik Akses Lambda Objek berbeda yang menginvokasi fungsi Lambda yang sama, sehingga memperluas fleksibilitas fungsi Lambda Anda.

> **Important**
>
> Saat menggunakan Titik Akses Lambda Objek, pastikan muatan tidak berisi informasi rahasia apa pun.

9. (Opsional) Untuk Rentang dan nomor bagian, Anda harus mengaktifkan opsi ini jika Anda ingin memproses permintaan `GET` dan `HEAD` dengan header rentang dan nomor bagian. Mengaktifkan opsi ini mengonfirmasi bahwa fungsi Lambda Anda dapat mengenali dan memproses permintaan ini. Untuk informasi lebih lanjut tentang berbagai header dan nomor bagian, lihat Bekerja dengan header Range dan partNumber.
10. (Opsional) Untuk Metrik permintaan, pilih Aktifkan atau Nonaktifkan untuk menambahkan pemantauan Amazon S3 ke Titik Akses Lambda Objek Anda. Metrik permintaan ditagih dengan tarif Amazon CloudWatch standar.
11. (Opsional) Di bawah Kebijakan Titik Akses Lambda, buat pengaturan kebijakan sumber daya. Kebijakan sumber daya memberikan izin untuk Titik Akses Lambda Objek yang ditentukan dan dapat mengontrol penggunaan titik akses berdasarkan sumber daya, pengguna, atau ketentuan lainnya. Untuk informasi selengkapnya tentang kebijakan sumber daya Titik Akses Lambda Object, lihat, Mengonfigurasi kebijakan IAM untuk Titik Akses Lambda Objek.
12. Di bagian bawah Pengaturan Blok Titik Akses untuk Titik Akses Lambda Objek ini, pilih pengaturan blokir akses publik yang ingin Anda terapkan. Semua pengaturan blokir akses publik diaktifkan secara default untuk Titik Akses Lambda Objek baru, dan kami menyarankan Anda membiarkan pengaturan default diaktifkan. Amazon S3 saat ini tidak mendukung perubahan pengaturan blokir akses publik milik Titik Akses Lambda Objek setelah Titik Akses Lambda Objek dibuat.

Untuk informasi selengkapnya tentang menggunakan Blokir Akses Publik Amazon S3., lihat Mengelola akses publik ke titik akses.

13. Pilih Buat Titik Akses Objek Lambda.

Menggunakan AWS CLI

Untuk membuat Object Lambda Access Point dengan menggunakan template AWS CloudFormation

> **Note**
>
> Untuk menggunakan perintah berikut ini, ganti *user input placeholders* dengan informasi Anda sendiri.

1. Unduh paket penerapan AWS Lambda fungsi `s3objectlambda_deployment_package.zip` di konfigurasi default Objek Lambda S3.
2. Jalankan perintah `put-object` berikut untuk mengunggah paket ke bucket Amazon S3.

```
aws s3api put-object --bucket Amazon S3 bucket name --key
 s3objectlambda_deployment_package.zip --body release/
s3objectlambda_deployment_package.zip
```

3. Unduh AWS CloudFormation template `s3objectlambda_defaultconfig.yaml` di konfigurasi default Objek Lambda S3.
4. Jalankan perintah `deploy` berikut untuk menyebarkan templat ke Akun AWS Anda.

```
aws cloudformation deploy --template-file s3objectlambda_defaultconfig.yaml \
 --stack-name AWS CloudFormation stack name \
 --parameter-overrides ObjectLambdaAccessPointName=Object Lambda Access Point name
 \
  SupportingAccessPointName=Amazon S3 access point S3BucketName=Amazon S3 bucket \
  LambdaFunctionS3BucketName=Amazon S3 bucket containing your Lambda package \
  LambdaFunctionS3Key=Lambda object key LambdaFunctionS3ObjectVersion=Lambda object
 version \
  LambdaFunctionRuntime=Lambda function runtime --capabilities capability_IAM
```

Anda dapat mengonfigurasi AWS CloudFormation template ini untuk memanggil Lambda `GET` untuk`HEAD`,, `LIST` dan operasi API. Untuk informasi selengkapnya tentang mengubah konfigurasi default templat, lihat the section called “Otomatiskan pengaturan Lambda Objek S3 dengan AWS CloudFormation”.

Untuk membuat Object Lambda Access Point dengan menggunakan AWS CLI

> **Note**
>
> Untuk menggunakan perintah berikut ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

Contoh berikut membuat Titik Akses Lambda Objek yang diberi nama *`my-object-lambda-ap`* untuk bucket *`DOC-EXAMPLE-BUCKET1`* di akun *`111122223333`*. Contoh ini mengasumsikan bahwa titik akses standar bernama *`example-ap`* telah dibuat. Untuk informasi selengkapnya tentang cara membuat titik akses standar, lihat the section called “Membuat titik akses”.

Contoh ini menggunakan fungsi AWS `decompress` prebuilt. Untuk informasi lebih lanjut tentang fungsi bawaan, lihat the section called “Menggunakan fungsi AWS yang dibangun”.

1. Buat bucket. Dalam contoh ini, kami akan menggunakan *`DOC-EXAMPLE-BUCKET1`*. Untuk informasi tentang membuat bucket, lihat the section called “Membuat bucket”.
2. Buat titik akses standar, dan lampirkan ke bucket Anda. Dalam contoh ini, kami akan menggunakan *`example-ap`*. Untuk informasi selengkapnya tentang cara membuat titik akses standar, lihat the section called “Membuat titik akses”.
3. Lakukan salah satu hal berikut ini:
   - Buat fungsi Lambda di akun Anda yang ingin Anda gunakan untuk mengubah objek Amazon S3 Anda. Untuk informasi selengkapnya tentang membuat fungsi Lambda, lihat the section called “Menulis fungsi Lambda”. Untuk menggunakan fungsi kustom Anda dengan AWS CLI, lihat Menggunakan Lambda dengan AWS CLI di Panduan AWS Lambda Pengembang.
   - Gunakan fungsi Lambda AWS bawaan. Untuk informasi lebih lanjut tentang fungsi bawaan, lihat Menggunakan fungsi Lambda yang AWS dibangun.
4. Buat file konfigurasi JSON bernama `my-olap-configuration.json`. Dalam konfigurasi ini, sediakan titik akses pendukung dan Amazon Resource Name (ARN) untuk fungsi Lambda yang Anda buat pada langkah sebelumnya atau ARN untuk fungsi bawaan yang Anda gunakan.

Example

```
{
    "SupportingAccessPoint" : "arn:aws:s3:us-
east-1:111122223333:accesspoint/example-ap",
    "TransformationConfigurations": [{
        "Actions" : ["GetObject", "HeadObject", "ListObjects", "ListObjectsV2"],
        "ContentTransformation" : {
            "AwsLambda": {
                "FunctionPayload" : "{\"compressionType\":\"gzip\"}",
                "FunctionArn" : "arn:aws:lambda:us-east-1:111122223333:function/
compress"
            }
        }
    }]
}
```

5. Jalankan perintah `create-access-point-for-object-lambda` untuk membuat Titik Akses Lambda Objek Anda.

```
aws s3control create-access-point-for-object-lambda --account-id 111122223333 --
name my-object-lambda-ap --configuration file://my-olap-configuration.json
```

6. (Opsional) Buat file kebijakan JSON bernama `my-olap-policy.json`.

   Menambahkan kebijakan sumber daya Titik Akses Lambda Objek dapat mengontrol penggunaan titik akses berdasarkan sumber daya, pengguna, atau kondisi lainnya. Kebijakan sumber daya ini memberikan izin akun `GetObject` *444455556666* ke Titik Akses Lambda Object yang telah ditentukan.

   Example

```
{
    "Version": "2008-10-17",
    "Statement": [
        {
            "Sid": "Grant account 444455556666 GetObject access",
            "Effect": "Allow",
            "Action": "s3-object-lambda:GetObject",
            "Principal": {
```

```
                "AWS": "arn:aws:iam::444455556666:root"
            },
            "Resource": "your-object-lambda-access-point-arn"
        }
    ]
}
```

7. (Opsional) Jalankan perintah `put-access-point-policy-for-object-lambda` untuk mengatur kebijakan sumber daya Anda.

```
aws s3control put-access-point-policy-for-object-lambda --account-id 111122223333
 --name my-object-lambda-ap --policy file://my-olap-policy.json
```

8. (Opsional) Tentukan muatan.

   Payload adalah JSON opsional yang dapat Anda berikan ke AWS Lambda fungsi Anda sebagai input. Anda dapat mengonfigurasi muatan dengan parameter berbeda untuk Titik Akses Lambda Objek berbeda yang menginvokasi fungsi Lambda yang sama, sehingga memperluas fleksibilitas fungsi Lambda Anda.

   Konfigurasi titik akses Lambda Objek berikut menunjukkan muatan dengan dua parameter.

```
{
 "SupportingAccessPoint": "AccessPointArn",
 "CloudWatchMetricsEnabled": false,
 "TransformationConfigurations": [{
  "Actions": ["GetObject", "HeadObject", "ListObjects", "ListObjectsV2"],
  "ContentTransformation": {
   "AwsLambda": {
    "FunctionArn": "FunctionArn",
    "FunctionPayload": "{\"res-x\": \"100\",\"res-y\": \"100\"}"
   }
  }
 }]
}
```

   Konfigurasi Titik Akses Lambda Objek berikut menunjukkan payload dengan satu parameter, dan dengan `GetObject-Range`, `GetObject-PartNumber`, `HeadObject-Range`, dan `HeadObject-PartNumber` diaktifkan.

```
{
```

```
    "SupportingAccessPoint":"AccessPointArn",
    "CloudWatchMetricsEnabled": false,
    "AllowedFeatures": ["GetObject-Range", "GetObject-PartNumber", "HeadObject-
Range", "HeadObject-PartNumber"],
    "TransformationConfigurations": [{
        "Action": ["GetObject", "HeadObject", "ListObjects", "ListObjectsV2"],
        "ContentTransformation": {
            "AwsLambda": {
                "FunctionArn":"FunctionArn",
                "FunctionPayload": "{\"compression-amount\": \"5\"}"
            }
        }
    }]
}
```

> **Important**
>
> Saat menggunakan Titik Akses Lambda Objek, pastikan muatan tidak berisi informasi rahasia apa pun.

Menggunakan AWS CloudFormation konsol dan template

Anda dapat membuat Titik Akses Lambda Objek dengan menggunakan konfigurasi default yang disediakan oleh Amazon S3. Anda dapat mengunduh AWS CloudFormation template dan kode sumber fungsi Lambda dari GitHub repositori dan menyebarkan sumber daya ini untuk menyiapkan Objek Lambda Access Point fungsional.

Untuk informasi tentang memodifikasi konfigurasi default AWS CloudFormation template, lihatthe section called “Otomatiskan pengaturan Lambda Objek S3 dengan AWS CloudFormation”.

Untuk informasi tentang mengonfigurasi Titik Akses Objek Lambda dengan AWS CloudFormation menggunakan tanpa templat, `AWS::S3ObjectLambda::AccessPoint`lihat di Panduan Pengguna AWS CloudFormation .

Untuk mengunggah paket deployment fungsi Lambda

1. Unduh paket penerapan AWS Lambda fungsi `s3objectlambda_deployment_package.zip` di konfigurasi default Objek Lambda S3.
2. Unggah paket ke bucket Amazon S3.

Untuk membuat Object Lambda Access Point dengan menggunakan konsol AWS CloudFormation

1. Unduh AWS CloudFormation template `s3objectlambda_defaultconfig.yaml` di konfigurasi default Objek Lambda S3.
2. Masuk ke AWS Management Console dan buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
3. Lakukan salah satu hal berikut ini:
   - Jika Anda belum pernah menggunakan AWS CloudFormation sebelumnya, di AWS CloudFormation beranda, pilih Buat tumpukan.
   - Jika Anda pernah menggunakan AWS CloudFormation sebelumnya, di panel navigasi kiri, pilih Tumpukan. Pilih Buat tumpukan, lalu pilih Dengan sumber daya baru (standar).
4. Untuk Prasyarat-Siapkan templat, pilih Templat sudah siap.
5. Untuk Tentukan templat, pilih Unggah file templat dan unggah`s3objectlambda_defaultconfig.yaml`.
6. Pilih Selanjutnya.
7. Pada halaman Tentukan detail tumpukan, masukkan nama untuk tumpukan.
8. Di bagian Parameter, tentukan parameter berikut, yang ditentukan di dalam templat tumpukan:
   a. Untuk CreateNewSupportingAccessPoint, lakukan salah satu hal berikut:
      - Jika Anda sudah memiliki titik akses pendukung untuk bucket S3 tempat Anda mengunggah templat, pilih false.
      - Jika Anda ingin membuat titik akses baru untuk bucket ini, pilih true.
   b. Untuk EnableCloudWatchMonitoring, pilih true atau false, tergantung apakah Anda ingin mengaktifkan metrik dan alarm CloudWatch permintaan Amazon.
   c. (Opsional) Untuk LambdaFunctionPayload, tambahkan teks JSON yang ingin Anda berikan ke fungsi Lambda Anda sebagai input. Anda dapat mengonfigurasi muatan dengan parameter berbeda untuk Titik Akses Lambda Objek berbeda yang menginvokasi fungsi Lambda yang sama, sehingga memperluas fleksibilitas fungsi Lambda Anda.

   > **Important**
   > Saat menggunakan Titik Akses Lambda Objek, pastikan muatan tidak berisi informasi rahasia apa pun.

d. Untuk LambdaFunctionRuntime, masukkan runtime pilihan Anda untuk fungsi Lambda. Pilihan yang tersedia adalah `nodejs14.x`, `python3.9`, `java11`.

e. Untuk LambdaFunctionS3 BucketName, masukkan nama bucket Amazon S3 tempat Anda mengunggah paket penerapan.

f. Untuk LambdaFunctionS3Key, masukkan kunci objek Amazon S3 tempat Anda mengunggah paket penerapan.

g. Untuk LambdaFunctionS3 ObjectVersion, masukkan versi objek Amazon S3 tempat Anda mengunggah paket penerapan.

h. Untuk ObjectLambdaAccessPointName, masukkan nama untuk Object Lambda Access Point Anda.

i. Untuk S3 BucketName, masukkan nama bucket Amazon S3 yang akan dikaitkan dengan Titik Akses Objek Lambda Anda.

j. Untuk SupportingAccessPointName, masukkan nama jalur akses pendukung Anda.

> **Note**
>
> Ini adalah titik akses yang terkait dengan bucket Amazon S3 yang Anda pilih pada langkah sebelumnya. Jika Anda tidak memiliki titik akses yang terkait dengan bucket Amazon S3 Anda, Anda dapat mengonfigurasi template untuk membuatnya untuk Anda dengan memilih true for. CreateNewSupportingAccessPoint

9. Pilih Berikutnya.
10. Pada Konfigurasikan halaman opsi stack, pilih Berikutnya.

    Untuk informasi selengkapnya tentang pengaturan opsional di halaman ini, lihat Mengatur AWS CloudFormation opsi tumpukan dalam AWS CloudFormation Panduan Pengguna.
11. Di halaman Tinjau, pilih Buat tumpukan.

Menggunakan AWS Cloud Development Kit (AWS CDK)

Untuk informasi selengkapnya tentang mengonfigurasi Titik Akses Objek Lambda menggunakan AWS CDK, *`AWS::S3ObjectLambda`*lihat Membangun Pustaka di AWS Cloud Development Kit (AWS CDK) Referensi API.

Otomatiskan pengaturan Lambda Objek S3 dengan templat CloudFormation

Anda dapat menggunakan AWS CloudFormation template untuk membuat Titik Akses Lambda Objek Amazon S3 dengan cepat. CloudFormationTemplate secara otomatis membuat sumber daya yang relevan, mengonfigurasi peran AWS Identity and Access Management (IAM), dan menyiapkan AWS Lambda fungsi yang secara otomatis menangani permintaan melalui Object Lambda Access Point. Dengan CloudFormation template, Anda dapat menerapkan praktik terbaik, meningkatkan postur keamanan Anda, dan mengurangi kesalahan yang disebabkan oleh proses manual.

GitHub Repositori ini berisi CloudFormation template dan kode sumber fungsi Lambda. Untuk petunjuk tentang cara menggunakan templat, lihat the section called “Membuat Titik Akses Objek Lambda”.

Fungsi Lambda yang disediakan dalam templat tidak menjalankan transformasi apa pun. Sebagai gantinya, ini mengembalikan objek Anda apa adanya dari bucket S3. Anda dapat mengkloning fungsi dan menambahkan kode transformasi Anda sendiri untuk mengubah dan memproses data saat dikembalikan ke aplikasi. Untuk informasi selengkapnya tentang fungsi Anda, lihat the section called “Memodifikasi fungsi Lambda” dan the section called “Menulis fungsi Lambda”.

Memodifikasi templat

Membuat titik akses pendukung baru

S3 Lambda Objek menggunakan dua titik akses, Titik Akses Lambda Objek dan titik akses S3 standar, yang disebut sebagai titik akses pendukung. Saat Anda membuat permintaan ke Titik Akses Lambda Objek, S3 akan memanggil Lambda atas nama Anda, atau mendelegasikan permintaan tersebut ke titik akses pendukung, bergantung pada konfigurasi Lambda Objek S3. Anda dapat membuat titik akses pendukung baru dengan meneruskan parameter berikut sebagai bagian dari perintah `aws cloudformation deploy` saat mendeploy templat.

```
CreateNewSupportingAccessPoint=true
```

Mengonfigurasi muatan fungsi

Anda dapat mengonfigurasi payload untuk menyediakan data tambahan ke fungsi Lambda dengan meneruskan parameter berikut sebagai bagian dari perintah `aws cloudformation deploy` saat menerapkan templat.

```
LambdaFunctionPayload="format=json"
```

Mengaktifkan pemantauan Amazon CloudWatch

Anda dapat mengaktifkan CloudWatch pemantauan dengan meneruskan parameter berikut sebagai bagian dari `aws cloudformation deploy` perintah saat menerapkan template.

```
EnableCloudWatchMonitoring=true
```

Parameter ini memungkinkan Titik Akses Objek Lambda Anda untuk metrik permintaan Amazon S3 dan membuat CloudWatch dua alarm untuk memantau kesalahan sisi klien dan sisi server.

> **Note**
>
> CloudWatch Penggunaan Amazon akan dikenakan biaya tambahan. Untuk informasi lebih lanjut tentang metrik permintaan Amazon S3, lihat Pemantauan dan pencatatan titik akses. Untuk detail harganya, lihat Harga CloudWatch .

Mengonfigurasi konkurensi yang tersedia

Untuk mengurangi latensi, Anda dapat mengonfigurasi konkurensi yang disediakan untuk fungsi Lambda yang mendukung Titik Akses Lambda Object dengan mengedit templat untuk menyertakan baris berikut di bawah `Resources`.

```
LambdaFunctionVersion:
      Type: AWS::Lambda::Version
      Properties:
        FunctionName: !Ref LambdaFunction
        ProvisionedConcurrencyConfig:
            ProvisionedConcurrentExecutions: Integer
```

> **Note**
>
> Anda akan dikenakan biaya tambahan untuk penyediaan konkurensi. Untuk informasi selengkapnya tentang konkurensi yang disediakan, lihat Mengelola konkurensi Lambda yang disediakan dalam Panduan Pengembang AWS Lambda .
> Untuk detail harganya, lihat Harga AWS Lambda.

Memodifikasi fungsi Lambda

Mengubah nilai header untuk permintaan **`GetObject`**

Secara default, fungsi Lambda meneruskan semua header, kecuali `Content-Length` dan`ETag`, dari permintaan URL yang telah ditetapkan sebelumnya ke `GetObject` klien. Berdasarkan kode transformasi Anda dalam fungsi Lambda, Anda dapat memilih untuk mengirim nilai header baru ke `GetObject` klien.

Anda dapat memperbarui fungsi Lambda Anda untuk mengirim nilai header baru dengan meneruskannya dalam operasi API `WriteGetObjectResponse`.

Misalnya, jika fungsi Lambda Anda menerjemahkan teks dalam objek Amazon S3 ke bahasa yang berbeda, Anda dapat meneruskan nilai baru di header `Content-Language`. Anda dapat melakukan ini dengan memodifikasi fungsi `writeResponse` sebagai berikut:

```
async function writeResponse (s3Client: S3, requestContext: GetObjectContext,
 transformedObject: Buffer,
  headers: Headers): Promise<PromiseResult<{}, AWSError>> {
  const { algorithm, digest } = getChecksum(transformedObject);

  return s3Client.writeGetObjectResponse({
    RequestRoute: requestContext.outputRoute,
    RequestToken: requestContext.outputToken,
    Body: transformedObject,
    Metadata: {
      'body-checksum-algorithm': algorithm,
      'body-checksum-digest': digest
    },
    ...headers,
    ContentLanguage: 'my-new-language'
  }).promise();
}
```

Untuk daftar lengkap header yang didukung, lihat `WriteGetObjectResponse` dalam Referensi API Amazon Simple Storage Service.

Mengembalikan header metadata

Anda dapat memperbarui fungsi Lambda Anda untuk mengirim nilai header baru dengan meneruskannya dalam permintaan operasi API `WriteGetObjectResponse`.

```
async function writeResponse (s3Client: S3, requestContext: GetObjectContext,
 transformedObject: Buffer,
  headers: Headers): Promise<PromiseResult<{}, AWSError>> {
  const { algorithm, digest } = getChecksum(transformedObject);

  return s3Client.writeGetObjectResponse({
    RequestRoute: requestContext.outputRoute,
    RequestToken: requestContext.outputToken,
    Body: transformedObject,
    Metadata: {
      'body-checksum-algorithm': algorithm,
      'body-checksum-digest': digest,
      'my-new-header': 'my-new-value'
    },
    ...headers
  }).promise();
}
```

Mengembalikan kode status baru

Anda dapat mengembalikan kode status kustom ke klien `GetObject` dengan meneruskannya dalam permintaan operasi API `WriteGetObjectResponse`.

```
async function writeResponse (s3Client: S3, requestContext: GetObjectContext,
 transformedObject: Buffer,
  headers: Headers): Promise<PromiseResult<{}, AWSError>> {
  const { algorithm, digest } = getChecksum(transformedObject);

  return s3Client.writeGetObjectResponse({
    RequestRoute: requestContext.outputRoute,
    RequestToken: requestContext.outputToken,
    Body: transformedObject,
    Metadata: {
      'body-checksum-algorithm': algorithm,
      'body-checksum-digest': digest
    },
    ...headers,
    StatusCode: Integer
  }).promise();
}
```

Untuk daftar lengkap kode status didukung, lihat `WriteGetObjectResponse` dalam Referensi API Amazon Simple Storage Service.

Menerapkan **`Range`** dan **`partNumber`** parameter ke objek sumber

Secara default, Object Lambda Access Point yang dibuat oleh CloudFormation template dapat menangani parameter `Range` dan`partNumber`. Fungsi Lambda menerapkan rentang atau nomor bagian yang diminta ke objek yang diubah. Untuk melakukannya, fungsi harus mengunduh seluruh objek dan menjalankan transformasi. Dalam beberapa kasus, rentang objek Anda yang diubah mungkin dipetakan tepat ke rentang objek sumber Anda. Artinya, meminta rentang byte A-B pada objek sumber dan menjalankan transformasi mungkin menghasilkan hasil yang sama seperti meminta seluruh objek, menjalankan transformasi, dan mengembalikan rentang byte A-B pada objek yang diubah.

Dalam kasus seperti itu, Anda dapat mengubah implementasi fungsi Lambda untuk menerapkan rentang atau nomor bagian langsung ke objek sumber. Pendekatan ini mengurangi latensi fungsi keseluruhan dan memori yang diperlukan. Untuk informasi selengkapnya, lihat the section called “Bekerja dengan header Range dan partNumber”.

Menonaktifkan penanganan **`Range`** dan **`partNumber`**

Secara default, Object Lambda Access Point yang dibuat oleh CloudFormation template dapat menangani parameter `Range` dan`partNumber`. Jika Anda tidak memerlukan perilaku ini, Anda dapat menonaktifkannya dengan menghapus baris berikut dari templat:

```
AllowedFeatures:
  - GetObject-Range
  - GetObject-PartNumber
  - HeadObject-Range
  - HeadObject-PartNumber
```

Mengubah objek yang berukuran besar

Secara default, fungsi Lambda memproses seluruh objek dalam memori sebelum dapat mulai streaming respons ke S3 Lambda Objek. Anda dapat memodifikasi fungsi untuk mengalirkan respons saat melakukan transformasi. Melakukannya akan membantu mengurangi latensi transformasi dan ukuran memori fungsi Lambda. Untuk implementasi contoh tersebut, lihat Contoh konten terkompresi Stream.

Menggunakan Titik Akses Lambda Objek Amazon S3

Membuat permintaan melalui Titik Akses Lambda Objek Amazon S3 berfungsi sama seperti membuat permintaan melalui titik akses lainnya. Untuk informasi selengkapnya tentang cara membuat permintaan melalui titik akses, lihat Menggunakan titik akses. Anda dapat membuat permintaan melalui Titik Akses Objek Lambda dengan menggunakan konsol Amazon S3 AWS Command Line Interface ,AWS CLI(), SDK AWS , atau Amazon S3 REST API.

> **Important**
>
> Amazon Resource Name (ARN) untuk Titik Akses Lambda Objek menggunakan nama layanan `s3-object-lambda`. Dengan demikian, Titik Akses Lambda Objek ARN dimulai dengan `arn:aws::s3-object-lambda`, bukan `arn:aws::s3`, yang digunakan dengan titik akses lainnya.

Cara untuk menemukan ARN untuk Titik Akses Lambda Objek Anda

Untuk menggunakan Object Lambda Access Point dengan AWS CLI atau AWS SDK, Anda perlu mengetahui Amazon Resource Name (ARN) dari Object Lambda Access Point. Contoh berikut menunjukkan cara untuk menemukan ARN untuk Titik Akses Lambda Objek dengan menggunakan konsol Amazon S3 atau. AWS CLI

Menggunakan konsol S3

Untuk menemukan ARN untuk Titik Akses Lambda Objek Anda dengan menggunakan konsol

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilih Titik Akses Lambda Objek.
3. Pilih tombol opsi di samping Titik Akses Lambda Objek yang ARN-nya ingin Anda salin.
4. Pilih Salin ARN.

Menggunakan AWS CLI

Untuk menemukan ARN untuk Object Lambda Access Point Anda dengan menggunakan AWS CLI

1. Untuk mengambil daftar Titik Akses Lambda Objek yang terkait dengan Akun AWS Anda, jalankan perintah berikut. Sebelum menjalankan perintah, ganti ID akun *111122223333*dengan Akun AWS ID Anda.

```
aws s3control list-access-points-for-object-lambda --account-id 111122223333
```

2. Tinjau output perintah untuk menemukan Titik Akses Lambda Objek ARN yang ingin Anda gunakan. Output dari perintah sebelumnya akan terlihat seperti contoh berikut.

```
{
    "ObjectLambdaAccessPointList": [
        {
            "Name": "my-object-lambda-ap",
            "ObjectLambdaAccessPointArn": "arn:aws:s3-object-lambda:us-
east-1:111122223333:accesspoint/my-object-lambda-ap"
        },
        ...
    ]
}
```

Cara menggunakan alias gaya bucket untuk bucket S3 Anda Titik Akses Lambda Objek

Saat Anda membuat Titik Akses Lambda Objek, Amazon S3 secara otomatis menghasilkan alias unik untuk Titik Akses Lambda Objek Anda. Anda dapat menggunakan alias ini alih-alih nama bucket Amazon S3 atau Amazon Resource Name (ARN) Titik Akses Lamda Objek dalam permintaan untuk operasi bidang data titik akses. Untuk mengetahui daftar operasi ini, lihat Kompatibilitas titik akses dengan AWS layanan.

Nama alias Titik Akses Lambda Objek dibuat dalam namespace yang sama dengan bucket Amazon S3. Nama alias ini dibuat secara otomatis, dan tidak dapat diubah. Untuk Titik Akses Lambda Objek yang ada, alias secara otomatis ditetapkan untuk digunakan. Nama alias Titik Akses Lambda Objek memenuhi semua persyaratan nama bucket Amazon S3 yang valid dan terdiri dari bagian-bagian berikut:

Object Lambda Access Point name prefix-*metadata*--ol-s3

> **Note**
>
> Sufiks `--ol-s3`dicadangkan untuk nama alias Titik Akses Lambda Objek dan tidak dapat digunakan untuk nama bucket atau Titik Akses Lambda Objek. Untuk informasi selengkapnya tentang penamaan bucket di Amazon S3, lihat Peraturan penamaan bucket.

Contoh berikut menunjukkan ARN dan alias Titik Akses Lambda Objek untuk Titik Akses Lambda Objek bernama *my-object-lambda-access-point*:

- ARN–`arn:aws:s3-object-lambda:`*region*`:`*account-id*`:accesspoint/`*my-object-lambda-access-point*
- Alias Titik Akses Lambda Objek–*my-object-lambda-acc-1a4n8yjrb3kda96f67zwrwiiuse1a*`--ol-s3`

Bila Anda menggunakan Titik Akses Lambda Objek, Anda dapat menggunakan nama alias Titik Akses Lambda Objek tanpa memerlukan perubahan kode yang ekstensif.

Ketika Anda menghapus Titik Akses Lambda Objek, nama alias Titik Akses Lambda Objek menjadi tidak aktif dan tidak tersedia.

Cara menemukan alias untuk Titik Akses Lambda Objek Anda

Menggunakan konsol S3

Untuk menemukan alias untuk Titik Akses Lambda Objek Anda dengan menggunakan konsol

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilih Titik Akses Lambda Objek.
3. Untuk Access Point Lambda Objek yang ingin Anda gunakan, salin nilai Alias Titik Akses Lambda Objek.

Menggunakan AWS CLI

Saat Anda membuat Titik Akses Lambda Objek, Amazon S3 secara otomatis menghasilkan nama alias Titik Akses Lambda Objek, seperti yang ditunjukkan pada perintah contoh berikut. Untuk

menjalankan perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri. Untuk informasi tentang cara membuat Titik Akses Objek Lambda menggunakan AWS CLI, lihat. Untuk membuat Object Lambda Access Point dengan menggunakan AWS CLI

```
aws s3control create-access-point-for-object-lambda --account-id 111122223333 --
name my-object-lambda-access-point --configuration file://my-olap-configuration.json
{
    "ObjectLambdaAccessPointArn": "arn:aws:s3:region:111122223333:accesspoint/my-
access-point",
    "Alias": {
        "Value": "my-object-lambda-acc-1a4n8yjrb3kda96f67zwrwiiuse1a--ol-s3",
        "Status": "READY"
    }
}
```

Nama alias Titik Akses Lambda Objek yang dihasilkan memiliki dua bidang:

- Kolom `Value` tersebut adalah nilai alias Titik Akses Lambda Objek.
- Bidang `Status` adalah status alias Titik Akses Lambda Objek. Jika statusnya adalah `PROVISIONING`, Amazon S3 menyediakan alias Titik Akses Lambda Objek, dan alias belum siap untuk digunakan. Jika statusnya adalah `READY`, alias Titik Akses Lambda Objek telah berhasil disediakan dan siap digunakan.

Untuk informasi selengkapnya tentang jenis data `ObjectLambdaAccessPointAlias` di API REST, lihat CreateAccessPointForObjectLambdadan ObjectLambdaAccessPointAlias dalam Referensi API Amazon Simple Storage Service.

Cara untuk menggunakan alias Titik Akses Lambda Objek alias

Anda dapat menggunakan alias Titik Akses Lambda Objek, bukan nama bucket Amazon S3 untuk operasi yang tercantum di dalam Kompatibilitas titik akses dengan AWS layanan.

AWS CLI Contoh berikut untuk `get-bucket-location` perintah menggunakan alias access point bucket untuk mengembalikan tempat Wilayah AWS bucket berada. Untuk menjalankan perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3api get-bucket-location --bucket my-object-lambda-
acc-w7i37nq6xuzgax3jw3oqtifiusw2a--ol-s3

{
```

```
    "LocationConstraint": "us-west-2"
}
```

Jika alias Titik Akses Lambda Object dalam permintaannya tidak valid, kode kesalahan `InvalidAccessPointAliasError` dikembalikan. Untuk informasi selengkapnya tentang `InvalidAccessPointAliasError`, lihat Daftar Kode Kesalahan dalam Referensi API Amazon Simple Storage Service.

Keterbatasan alias Titik Akses Lambda Objek sama dengan alias titik akses. Untuk informasi lebih lanjut tentang batasan alias titik akses, lihat Batasan.

Pertimbangan keamanan untuk Titik Akses S3 Lambda Objek

Dengan Amazon S3 Object Lambda, Anda dapat melakukan transformasi kustom pada data saat meninggalkan Amazon S3 dengan menggunakan skala dan fleksibilitas sebagai platform komputasi. AWS Lambda S3 dan Lambda tetap aman secara default, namun untuk menjaga keamanan ini, pertimbangan kustom oleh pembuat fungsi Lambda diperlukan. S3 Lambda Objek mengharuskan semua akses dilakukan oleh pengguna utama yang diautentikasi (tidak ada akses anonim) dan melalui HTTPS.

Untuk memitigasi risiko keamanan, kami merekomendasikan hal berikut:

- Cakup peran eksekusi Lambda ke kumpulan izin sekecil mungkin.
- Jika memungkinkan, pastikan fungsi Lambda Anda mengakses Amazon S3 melalui URL yang telah ditentukan sebelumnya.

Mengonfigurasi kebijakan IAM

Kebijakan sumber daya dukungan titik akses S3 AWS Identity and Access Management (IAM) yang memungkinkan Anda mengontrol penggunaan titik akses berdasarkan sumber daya, pengguna, atau kondisi lainnya. Untuk informasi selengkapnya, lihat Mengonfigurasi kebijakan IAM untuk Titik Akses Lambda Objek.

Perilaku enkripsi

Karena Titik Akses Objek Lambda menggunakan Amazon S3 AWS Lambda dan, ada perbedaan dalam perilaku enkripsi. Untuk informasi selengkapnya tentang enkripsi default S3, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3.

- Saat Anda menggunakan enkripsi di sisi server S3 dengan Titik Akses Lambda Object, objek didekripsi sebelum dikirimkan ke Lambda. Setelah objek dikirim ke Lambda, objek diproses secara tidak terenkripsi (dalam kasus permintaan `GET` atau `HEAD`).
- Untuk mencegah kunci enkripsi dicatat, S3 menolak permintaan `GET` dan `HEAD` untuk objek yang dienkripsi menggunakan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C). Namun, fungsi Lambda mungkin masih mengambil objek ini jika memiliki akses ke kunci klien yang disediakan.
- Saat menggunakan enkripsi di sisi klien S3 dengan Titik Akses Lambda Objeks, pastikan Lambda memiliki akses ke kunci enkripsi sehingga dapat mendekripsi dan mengenkripsi kembali objek.

Keamanan titik akses

S3 Lambda Objek menggunakan dua titik akses, Titik Akses Lambda Objek dan titik akses S3 standar, yang disebut sebagai titik akses pendukung. Saat Anda membuat permintaan ke Titik Akses Lambda Objek, S3 akan memanggil Lambda atas nama Anda, atau mendelegasikan permintaan tersebut ke titik akses pendukung, bergantung pada konfigurasi Lambda Objek S3. Ketika Lambda dipanggil untuk permintaan, S3 menghasilkan URL yang telah ditandatangani sebelumnya ke objek Anda atas nama Anda melalui titik akses pendukung. Fungsi Lambda Anda menerima URL ini sebagai input saat fungsi tersebut dipanggil.

Anda dapat mengatur fungsi Lambda Anda untuk menggunakan URL presigned ini untuk mengambil objek asli, alih-alih memanggil S3 secara langsung. Dengan menggunakan model ini, Anda dapat menerapkan batas keamanan yang lebih baik ke objek Anda. Anda dapat membatasi akses objek langsung melalui bucket S3 atau titik akses S3 ke serangkaian peran IAM atau pengguna terbatas. Pendekatan ini juga melindungi fungsi Lambda Anda agar tidak terkena masalah confused deputy, di mana fungsi yang salah dikonfigurasi dengan izin berbeda dari pemanggil dapat mengizinkan atau menolak akses ke objek, padahal seharusnya tidak demikian.

Akses publik Titik Akses Lambda Objek

S3 Lambda Objek tidak mengizinkan akses anonim atau publik karena Amazon S3 harus mengotorisasi identitas Anda untuk menyelesaikan permintaan S3 Lambda Objek. Saat memanggil permintaan melalui Titik Akses Lambda Objek, Anda harus memiliki izin untuk fungsi Lambda `lambda:InvokeFunction` yang dikonfigurasi. Demikian pula, saat menjalankan operasi API lainnya melalui Titik Akses Lambda Objek, Anda harus memiliki izin `s3:*` yang diperlukan.

Tanpa izin ini, permintaan untuk menginvokasi Lambda atau mendelegasikan ke S3 akan gagal dengan kesalahan HTTP 403 (Forbidden). Semua akses harus dilakukan oleh pengguna utama

yang diautentikasi. Jika Anda memerlukan akses publik, Anda dapat menggunakan Lambda @Edge sebagai alternatif yang memungkinkan. Untuk informasi selengkapnya, lihat Menyesuaikan di tepi dengan Lambda @Edge di Panduan Pengembang CloudFront Amazon.

Alamat IP Titik Akses Lambda Objek

Subnet `describe-managed-prefix-lists` mendukung titik akhir gateway cloud privat virtual (VPC), dan terkait dengan tabel perutean titik akhir VPC. Karena Titik Akses Objek Lambda tidak mendukung VPC gateway, rentang IP-nya hilang. Rentang yang hilang adalah milik Amazon S3, namun tidak didukung oleh titik akhir VPC gateway. Untuk informasi selengkapnya`describe-managed-prefix-lists`, lihat DescribeManagedPrefixListsdi Referensi API Amazon EC2 dan rentang alamat AWS IP di. Referensi Umum AWS

Mengonfigurasi kebijakan IAM untuk Titik Akses Lambda Objek

Kebijakan sumber daya dukungan titik akses Amazon S3 AWS Identity and Access Management (IAM) yang dapat Anda gunakan untuk mengontrol penggunaan titik akses menurut sumber daya, pengguna, atau kondisi lainnya. Anda dapat mengontrol akses melalui kebijakan sumber daya opsional pada Titik Akses Lambda Objek, atau kebijakan sumber daya untuk mendukung titik akses. Sebagai step-by-step contoh, lihat Tutorial: Mengubah data untuk aplikasi Anda dengan S3 Object Lambda danTutorial: Mendeteksi dan menyunting data PII dengan S3 Object Lambda dan Amazon Comprehend.

Empat sumber daya berikut harus memiliki izin yang diberikan untuk bekerja dengan Titik Akses Lambda Objek:

- Identitas IAM, seperti pengguna atau peran. Untuk informasi selengkapnya tentang identitas IAM dan praktik terbaik, lihat Identitas IAM (pengguna, grup pengguna, dan peran) dalam Panduan Pengguna IAM.
- Bucket dan titik akses standar terkait. Saat Anda bekerja dengan Titik Akses Lambda Objek, titik akses standar ini dikenal sebagai titik akses pendukung.
- Titik akses Lambda Object.
- AWS Lambda Fungsinya.

> **Important**
>
> Sebelum menyimpan kebijakan, pastikan untuk menyelesaikan peringatan keamanan, kesalahan, peringatan umum, dan saran dari. AWS Identity and Access Management Access

Analyzer Penganalisis Akses IAM menjalankan pemeriksaan kebijakan untuk memvalidasi kebijakan Anda terhadap tata bahasa kebijakan IAM dan praktik terbaik. Pemeriksaan ini menghasilkan temuan dan memberikan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang fungsional dan sesuai dengan praktik terbaik keamanan.
Untuk mempelajari validasi kebijakan menggunakan Penganalisis Akses IAM lebih lanjut, lihat Memvalidasi kebijakan Penganalisis Akses IAM di Panduan Pengguna IAM. Untuk melihat daftar peringatan, kesalahan, dan saran yang ditampilkan oleh Penganalisis Akses IAM, lihat referensi pemeriksaan kebijakan Penganalisis Akses IAM.

Contoh kebijakan berikut mengasumsikan bahwa Anda memiliki sumber daya berikut:

- Bucket Amazon S3 dengan Amazon Resource Name (ARN) berikut:

  `arn:aws:s3:::DOC-EXAMPLE-BUCKET1`
- Titik akses Amazon S3 standard pada bucket ini dengan ARN berikut:

  `arn:aws:s3:us-east-1:111122223333:accesspoint/my-access-point`
- Titik akses Lambda Objek dengan ARN berikut:

  `arn:aws:s3-object-lambda:us-east-1:111122223333:accesspoint/my-object-lambda-ap`
- AWS Lambda Fungsi dengan ARN berikut:

  `arn:aws:lambda:us-east-1:111122223333:function:MyObjectLambdaFunction`

> **Note**
>
> Jika Anda menggunakan fungsi Lambda dari akun, Anda harus menyertakan versi fungsi tertentu dalam pernyataan kebijakan Anda. *Dalam contoh ARN berikut, versi ditunjukkan oleh 1:*
> `arn:aws:lambda:us-east-1:111122223333:function:MyObjectLambdaFunction:1`
> Lambda tidak mendukung penambahan kebijakan IAM ke versi. `$LATEST` Untuk informasi selengkapnya tentang versi fungsi Lambda, lihat Versi fungsi Lambda dalam Panduan Pengembang AWS Lambda .

Example – Kebijakan bucket yang mendelegasikan kontrol akses ke titik akses standar

Contoh kebijakan bucket S3 berikut mendelegasikan kontrol akses untuk sebuah bucket ke titik akses standar bucket. Kebijakan ini memungkinkan akses penuh ke semua titik akses yang dimiliki oleh akun pemilik bucket. Oleh karena itu, semua akses ke bucket ini dikendalikan oleh kebijakan yang melekat pada titik aksesnya. Pengguna dapat membaca dari bucket hanya melalui titik akses, yang berarti bahwa operasi hanya dapat dipanggil melalui titik akses. Untuk informasi selengkapnya, lihat Mendelegasikan kontrol akses ke titik akses.

```
{
    "Version": "2012-10-17",
    "Statement" : [
    {
        "Effect": "Allow",
        "Principal" : { "AWS":"account-ARN"},
        "Action" : "*",
        "Resource" : [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET1",
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
        ],
        "Condition": {
            "StringEquals" : { "s3:DataAccessPointAccount" : "Bucket owner's account
 ID" }
        }
    }]
}
```

Example — Kebijakan IAM yang memberikan pengguna izin yang diperlukan untuk menggunakan Titik Akses Objek Lambda

Kebijakan IAM berikut memberikan izin pengguna ke fungsi Lambda, titik akses standar, dan Titik Akses Lambda Objek.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowLambdaInvocation",
      "Action": [
        "lambda:InvokeFunction"
      ],
      "Effect": "Allow",
```

```
      "Resource": "arn:aws:lambda:us-
east-1:111122223333:function:MyObjectLambdaFunction:1",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": [
            "s3-object-lambda.amazonaws.com"
          ]
        }
      }
    },
    {
      "Sid": "AllowStandardAccessPointAccess",
      "Action": [
        "s3:Get*",
        "s3:List*"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:us-east-1:111122223333:accesspoint/my-access-point/*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": [
            "s3-object-lambda.amazonaws.com"
          ]
        }
      }
    },
    {
      "Sid": "AllowObjectLambdaAccess",
      "Action": [
        "s3-object-lambda:Get*",
        "s3-object-lambda:List*"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3-object-lambda:us-east-1:111122223333:accesspoint/my-
object-lambda-ap"
    }
  ]
}
```

Aktifkan izin untuk peran eksekusi Lambda

Ketika `GET` permintaan dibuat ke Object Lambda Access Point, fungsi Lambda Anda memerlukan izin untuk mengirim data ke S3 Object Lambda Access Point. Izin ini diberikan dengan mengaktifkan

izin `s3-object-lambda:WriteGetObjectResponse` pada peran eksekusi fungsi Lambda Anda. Anda dapat membuat peran eksekusi baru atau memperbarui peran yang sudah ada.

> **Note**
>
> Fungsi Anda memerlukan izin `s3-object-lambda:WriteGetObjectResponse` hanya jika Anda membuat permintaan `GET`.

Untuk membuat peran eksekusi di konsol IAM

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.
3. Pilih Buat peran.
4. Di bawah Kasus penggunaan umum, pilih Lambda.
5. Pilih Selanjutnya.
6. Pada halaman Tambahkan izin, cari kebijakan AWS terkelola `AmazonS3ObjectLambdaExecutionRolePolicy`, lalu pilih kotak centang di samping nama kebijakan.

   Kebijakan ini harus berisi Tindakan `s3-object-lambda:WriteGetObjectResponse`.
7. Pilih Selanjutnya.
8. Pada halaman Nama, tinjau, dan buat, untuk Nama peran, masukkan **`s3-object-lambda-role`**.
9. (Opsional) Tambahkan deskripsi dan tag untuk peran ini.
10. Pilih Buat peran.
11. Terapkan **`s3-object-lambda-role`** yang baru dibuat sebagai peran eksekusi fungsi Lambda Anda. Ini dapat dilakukan selama atau setelah pembuatan fungsi Lambda di konsol Lambda.

Untuk informasi selengkapnya tentang peran eksekusi, lihat Peran eksekusi Lambda di Panduan Pengembang AWS Lambda .

Menggunakan kunci konteks dengan Titik Akses Lambda Objek

S3 Lambda Objek akan mengevaluasi kunci konteks seperti `s3-object-lambda:TlsVersion` atau `s3-object-lambda:AuthType` yang terkait dengan koneksi atau penandatanganan permintaan. Semua kunci konteks lainnya, seperti `s3:prefix`, dievaluasi oleh Amazon S3.

Dukungan CORS Titik Akses Lambda Objek

Ketika S3 Lambda Objek menerima permintaan dari browser atau permintaannya menyertakan header `Origin`, S3 Lambda Objek selalu menambahkan bidang header `“AllowedOrigins":"*"`.

Untuk informasi selengkapnya, lihat Berbagi sumber daya lintas asal (CORS).

Menulis fungsi Lambda untuk Titik Akses S3 Lambda Objek

Bagian ini merinci cara menulis AWS Lambda fungsi untuk digunakan dengan Amazon S3 Object Lambda Access Points.

Untuk mempelajari tentang end-to-end prosedur lengkap untuk beberapa tugas Lambda Objek S3, lihat berikut ini:

- Tutorial: Mengubah data untuk aplikasi Anda dengan S3 Object Lambda
- Tutorial: Mendeteksi dan menyunting data PII dengan S3 Object Lambda dan Amazon Comprehend
- Tutorial: Menggunakan S3 Lambda Objek untuk menandai gambar secara dinamis saat gambarnya diambil

Topik

- Bekerja dengan GetObject permintaan di Lambda
- Bekerja dengan permintaan HeadObject di Lambda
- Bekerja dengan permintaan ListObjects di Lambda
- Bekerja dengan permintaan ListObjectsV2 di Lambda
- Format konteks peristiwa dan penggunaan
- Bekerja dengan header Range dan partNumber

Bekerja dengan **`GetObject`** permintaan di Lambda

Bagian ini mengasumsikan bahwa Titik Akses Lambda Objek Anda dikonfigurasi untuk memanggil fungsi Lambda `GetObject`. S3 Lambda Objek mencakup operasi API Amazon S3, `WriteGetObjectResponse`, yang mengaktifkan fungsi Lambda untuk memberikan data yang disesuaikan dan respons header untuk pemanggil `GetObject`.

`WriteGetObjectResponse` memberikan Anda kontrol ekstensif atas kode status, header respons, dan badan respons, berdasarkan kebutuhan pemrosesan Anda. Anda dapat menggunakan `WriteGetObjectResponse` untuk menanggapi dengan seluruh objek yang berubah, bagian dari objek yang berubah, atau tanggapan lain berdasarkan konteks aplikasi Anda. Bagian berikut menampilkan contoh-contoh unik penggunaan operasi API `WriteGetObjectResponse`.

- Contoh 1: Menanggapi dengan kode status HTTP 403 (Forbidden)
- Contoh 2: Merespons dengan citra yang berubah
- Contoh 3: Melakukan streaming konten terkompresi

Contoh 1: Menanggapi dengan kode status HTTP 403 (Forbidden)

Anda dapat menggunakan `WriteGetObjectResponse` untuk menanggapi dengan kode status HTTP 403 (Terlarang) berdasarkan isi dari objek.

Java

```
package com.amazon.s3.objectlambda;

import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.events.S3ObjectLambdaEvent;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3Client;
import com.amazonaws.services.s3.model.WriteGetObjectResponseRequest;

import java.io.ByteArrayInputStream;
import java.net.URI;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;

public class Example1 {
```

```
    public void handleRequest(S3ObjectLambdaEvent event, Context context) throws
 Exception {
        AmazonS3 s3Client = AmazonS3Client.builder().build();

        // Check to see if the request contains all of the necessary information.
        // If it does not, send a 4XX response and a custom error code and message.
        // Otherwise, retrieve the object from S3 and stream it
        // to the client unchanged.
        var tokenIsNotPresent = !
event.getUserRequest().getHeaders().containsKey("requiredToken");
        if (tokenIsNotPresent) {
            s3Client.writeGetObjectResponse(new WriteGetObjectResponseRequest()
                    .withRequestRoute(event.outputRoute())
                    .withRequestToken(event.outputToken())
                    .withStatusCode(403)
                    .withContentLength(0L).withInputStream(new
 ByteArrayInputStream(new byte[0]))
                    .withErrorCode("MissingRequiredToken")
                    .withErrorMessage("The required token was not present in the
 request."));
            return;
        }

        // Prepare the presigned URL for use and make the request to S3.
        HttpClient httpClient = HttpClient.newBuilder().build();
        var presignedResponse = httpClient.send(
                HttpRequest.newBuilder(new URI(event.inputS3Url())).GET().build(),
                HttpResponse.BodyHandlers.ofInputStream());

        // Stream the original bytes back to the caller.
        s3Client.writeGetObjectResponse(new WriteGetObjectResponseRequest()
                .withRequestRoute(event.outputRoute())
                .withRequestToken(event.outputToken())
                .withInputStream(presignedResponse.body()));
    }
}
```

Python

```
import boto3
import requests

def handler(event, context):
```

```
    s3 = boto3.client('s3')

    """
    Retrieve the operation context object from the event. This object indicates
 where the WriteGetObjectResponse request
    should be delivered and contains a presigned URL in 'inputS3Url' where we can
 download the requested object from.
    The 'userRequest' object has information related to the user who made this
 'GetObject' request to
    S3 Object Lambda.
    """
    get_context = event["getObjectContext"]
    user_request_headers = event["userRequest"]["headers"]

    route = get_context["outputRoute"]
    token = get_context["outputToken"]
    s3_url = get_context["inputS3Url"]

    # Check for the presence of a 'CustomHeader' header and deny or allow based on
 that header.
    is_token_present = "SuperSecretToken" in user_request_headers

    if is_token_present:
        # If the user presented our custom 'SuperSecretToken' header, we send the
 requested object back to the user.
        response = requests.get(s3_url)
        s3.write_get_object_response(RequestRoute=route, RequestToken=token,
 Body=response.content)
    else:
        # If the token is not present, we send an error back to the user.
        s3.write_get_object_response(RequestRoute=route, RequestToken=token,
 StatusCode=403,
        ErrorCode="NoSuperSecretTokenFound", ErrorMessage="The request was not
 secret enough.")

    # Gracefully exit the Lambda function.
    return { 'status_code': 200 }
```

Node.js

```
const { S3 } = require('aws-sdk');
const axios = require('axios').default;
```

```
exports.handler = async (event) => {
    const s3 = new S3();

    // Retrieve the operation context object from the event. This object indicates
 where the WriteGetObjectResponse request
    // should be delivered and contains a presigned URL in 'inputS3Url' where we can
 download the requested object from.
    // The 'userRequest' object has information related to the user who made this
 'GetObject' request to S3 Object Lambda.
    const { userRequest, getObjectContext } = event;
    const { outputRoute, outputToken, inputS3Url } = getObjectContext;

    // Check for the presence of a 'CustomHeader' header and deny or allow based on
 that header.
    const isTokenPresent = Object
        .keys(userRequest.headers)
        .includes("SuperSecretToken");

    if (!isTokenPresent) {
        // If the token is not present, we send an error back to the user. The
 'await' in front of the request
        // indicates that we want to wait for this request to finish sending before
 moving on.
        await s3.writeGetObjectResponse({
            RequestRoute: outputRoute,
            RequestToken: outputToken,
            StatusCode: 403,
            ErrorCode: "NoSuperSecretTokenFound",
            ErrorMessage: "The request was not secret enough.",
        }).promise();
    } else {
        // If the user presented our custom 'SuperSecretToken' header, we send the
 requested object back to the user.
        // Again, note the presence of 'await'.
        const presignedResponse = await axios.get(inputS3Url);
        await s3.writeGetObjectResponse({
            RequestRoute: outputRoute,
            RequestToken: outputToken,
            Body: presignedResponse.data,
        }).promise();
    }

    // Gracefully exit the Lambda function.
    return { statusCode: 200 };
```

```
}
```

Contoh 2: Merespons dengan citra yang berubah

Saat melakukan transformasi citra, Anda mungkin memerlukan semua byte objek sumber sebelum Anda dapat mulai memprosesnya. Dalam kasus ini, permintaan `WriteGetObjectResponse` Anda akan mengembalikan seluruh objek ke aplikasi permintaan dalam satu panggilan.

Java

```
package com.amazon.s3.objectlambda;

import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.events.S3ObjectLambdaEvent;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3Client;
import com.amazonaws.services.s3.model.WriteGetObjectResponseRequest;

import javax.imageio.ImageIO;
import java.awt.image.BufferedImage;
import java.awt.Image;
import java.io.ByteArrayInputStream;
import java.io.ByteArrayOutputStream;
import java.net.URI;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;

public class Example2 {

    private static final int HEIGHT = 250;
    private static final int WIDTH = 250;

    public void handleRequest(S3ObjectLambdaEvent event, Context context) throws
 Exception {
        AmazonS3 s3Client = AmazonS3Client.builder().build();
        HttpClient httpClient = HttpClient.newBuilder().build();

        // Prepare the presigned URL for use and make the request to S3.
        var presignedResponse = httpClient.send(
                HttpRequest.newBuilder(new URI(event.inputS3Url())).GET().build(),
                HttpResponse.BodyHandlers.ofInputStream());
```

```
        // The entire image is loaded into memory here so that we can resize it.
        // Once the resizing is completed, we write the bytes into the body
        // of the WriteGetObjectResponse request.
        var originalImage = ImageIO.read(presignedResponse.body());
        var resizingImage = originalImage.getScaledInstance(WIDTH, HEIGHT,
 Image.SCALE_DEFAULT);
        var resizedImage = new BufferedImage(WIDTH, HEIGHT,
 BufferedImage.TYPE_INT_RGB);
        resizedImage.createGraphics().drawImage(resizingImage, 0, 0, WIDTH, HEIGHT,
 null);

        var baos = new ByteArrayOutputStream();
        ImageIO.write(resizedImage, "png", baos);

        // Stream the bytes back to the caller.
        s3Client.writeGetObjectResponse(new WriteGetObjectResponseRequest()
                .withRequestRoute(event.outputRoute())
                .withRequestToken(event.outputToken())
                .withInputStream(new ByteArrayInputStream(baos.toByteArray())));
    }
}
```

Python

```
import boto3
import requests
import io
from PIL import Image

def handler(event, context):
    """
    Retrieve the operation context object from the event. This object indicates
 where the WriteGetObjectResponse request
    should be delivered and has a presigned URL in 'inputS3Url' where we can
 download the requested object from.
    The 'userRequest' object has information related to the user who made this
 'GetObject' request to
    S3 Object Lambda.
    """
    get_context = event["getObjectContext"]
    route = get_context["outputRoute"]
    token = get_context["outputToken"]
```

```
    s3_url = get_context["inputS3Url"]

    """
    In this case, we're resizing .png images that are stored in S3 and are
 accessible through the presigned URL
    'inputS3Url'.
    """
    image_request = requests.get(s3_url)
    image = Image.open(io.BytesIO(image_request.content))
    image.thumbnail((256,256), Image.ANTIALIAS)

    transformed = io.BytesIO()
    image.save(transformed, "png")

    # Send the resized image back to the client.
    s3 = boto3.client('s3')
    s3.write_get_object_response(Body=transformed.getvalue(), RequestRoute=route,
 RequestToken=token)

    # Gracefully exit the Lambda function.
    return { 'status_code': 200 }
```

Node.js

```
const { S3 } = require('aws-sdk');
const axios = require('axios').default;
const sharp = require('sharp');

exports.handler = async (event) => {
    const s3 = new S3();

    // Retrieve the operation context object from the event. This object indicates
 where the WriteGetObjectResponse request
    // should be delivered and has a presigned URL in 'inputS3Url' where we can
 download the requested object from.
    const { getObjectContext } = event;
    const { outputRoute, outputToken, inputS3Url } = getObjectContext;

    // In this case, we're resizing .png images that are stored in S3 and are
 accessible through the presigned URL
    // 'inputS3Url'.
    const { data } = await axios.get(inputS3Url, { responseType: 'arraybuffer' });
```

```
    // Resize the image.
    const resized = await sharp(data)
        .resize({ width: 256, height: 256 })
        .toBuffer();

    // Send the resized image back to the client.
    await s3.writeGetObjectResponse({
        RequestRoute: outputRoute,
        RequestToken: outputToken,
        Body: resized,
    }).promise();

    // Gracefully exit the Lambda function.
    return { statusCode: 200 };
}
```

Contoh 3: Melakukan streaming konten terkompresi

Saat Anda mengompresi objek, data terkompresi dihasilkan secara bertahap. Akibatnya, Anda dapat menggunakan permintaan `WriteGetObjectResponse` Anda untuk mengembalikan data terkompresi segera setelah semuanya siap. Seperti yang ditunjukkan dalam contoh ini, Anda tidak perlu mengetahui panjang transformasi yang telah selesai.

Java

```
package com.amazon.s3.objectlambda;

import com.amazonaws.services.lambda.runtime.events.S3ObjectLambdaEvent;
import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3Client;
import com.amazonaws.services.s3.model.WriteGetObjectResponseRequest;

import java.net.URI;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;

public class Example3 {
```

```
    public void handleRequest(S3ObjectLambdaEvent event, Context context) throws
 Exception {
        AmazonS3 s3Client = AmazonS3Client.builder().build();
        HttpClient httpClient = HttpClient.newBuilder().build();

        // Request the original object from S3.
        var presignedResponse = httpClient.send(
                HttpRequest.newBuilder(new URI(event.inputS3Url())).GET().build(),
                HttpResponse.BodyHandlers.ofInputStream());

        // Consume the incoming response body from the presigned request,
        // apply our transformation on that data, and emit the transformed bytes
        // into the body of the WriteGetObjectResponse request as soon as they're
 ready.
        // This example compresses the data from S3, but any processing pertinent
        // to your application can be performed here.
        var bodyStream = new GZIPCompressingInputStream(presignedResponse.body());

        // Stream the bytes back to the caller.
        s3Client.writeGetObjectResponse(new WriteGetObjectResponseRequest()
                .withRequestRoute(event.outputRoute())
                .withRequestToken(event.outputToken())
                .withInputStream(bodyStream));
    }

}
```

Python

```
import boto3
import requests
import zlib
from botocore.config import Config


"""
A helper class to work with content iterators. Takes an interator and compresses the
 bytes that come from it. It
implements 'read' and '__iter__' so that the SDK can stream the response.
"""
class Compress:
    def __init__(self, content_iter):
        self.content = content_iter
```

```
        self.compressed_obj = zlib.compressobj()

    def read(self, _size):
        for data in self.__iter__()
            return data

    def __iter__(self):
        while True:
            data = next(self.content)
            chunk = self.compressed_obj.compress(data)
            if not chunk:
                break

            yield chunk

        yield self.compressed_obj.flush()


def handler(event, context):
    """
    Setting the 'payload_signing_enabled' property to False allows us to send a
 streamed response back to the client.
    in this scenario, a streamed response means that the bytes are not buffered into
 memory as we're compressing them,
    but instead are sent straight to the user.
    """
    my_config = Config(
        region_name='eu-west-1',
        signature_version='s3v4',
        s3={
            "payload_signing_enabled": False
        }
    )
    s3 = boto3.client('s3', config=my_config)

    """
    Retrieve the operation context object from the event. This object indicates
 where the WriteGetObjectResponse request
    should be delivered and has a presigned URL in 'inputS3Url' where we can
 download the requested object from.
    The 'userRequest' object has information related to the user who made this
 'GetObject' request to S3 Object Lambda.
    """
    get_context = event["getObjectContext"]
```

```
    route = get_context["outputRoute"]
    token = get_context["outputToken"]
    s3_url = get_context["inputS3Url"]

    # Compress the 'get' request stream.
    with requests.get(s3_url, stream=True) as r:
        compressed = Compress(r.iter_content())

        # Send the stream back to the client.
        s3.write_get_object_response(Body=compressed, RequestRoute=route,
 RequestToken=token, ContentType="text/plain",
                                     ContentEncoding="gzip")

    # Gracefully exit the Lambda function.
    return {'status_code': 200}
```

Node.js

```
const { S3 } = require('aws-sdk');
const axios = require('axios').default;
const zlib = require('zlib');

exports.handler = async (event) => {
    const s3 = new S3();

    // Retrieve the operation context object from the event. This object indicates
 where the WriteGetObjectResponse request
    // should be delivered and has a presigned URL in 'inputS3Url' where we can
 download the requested object from.
    const { getObjectContext } = event;
    const { outputRoute, outputToken, inputS3Url } = getObjectContext;

    // Download the object from S3 and process it as a stream, because it might be a
 huge object and we don't want to
    // buffer it in memory. Note the use of 'await' because we want to wait for
 'writeGetObjectResponse' to finish
    // before we can exit the Lambda function.
    await axios({
        method: 'GET',
        url: inputS3Url,
        responseType: 'stream',
    }).then(
        // Gzip the stream.
```

```
        response => response.data.pipe(zlib.createGzip())
    ).then(
        // Finally send the gzip-ed stream back to the client.
        stream => s3.writeGetObjectResponse({
            RequestRoute: outputRoute,
            RequestToken: outputToken,
            Body: stream,
            ContentType: "text/plain",
            ContentEncoding: "gzip",
        }).promise()
    );

    // Gracefully exit the Lambda function.
    return { statusCode: 200 };
}
```

> **Note**
>
> Meskipun S3 Lambda Objek mengizinkan hingga 60 detik untuk mengirim respons lengkap kepada pemanggil melalui permintaan `WriteGetObjectResponse`, jumlah aktual waktu yang tersedia mungkin kurang. Misalnya, batas waktu fungsi Lambda Anda kemungkinan kurang dari 60 detik. Dalam kasus lain, pemanggil mungkin memiliki batas waktu yang lebih ketat.

Agar penelepon asli menerima respons selain kode status HTTP 500 (Internal Server Error), panggilan `WriteGetObjectResponse` harus diselesaikan. Jika fungsi Lambda kembali, dengan pengecualian atau sebaliknya, sebelum operasi API `WriteGetObjectResponse` dipanggil, pemanggil asli menerima respons 500 (Internal Server Error). Pengecualian yang diberikan selama waktu yang diperlukan untuk menyelesaikan respons akan mengakibatkan respons terpotong kepada pemanggil. Jika fungsi Lambda menerima respons kode status HTTP 200 (OK) dari panggilan API `WriteGetObjectResponse`, lalu pemanggil asli telah mengirim permintaan lengkap. Respon fungsi Lambda, terlepas dari pengecualian diberikan atau tidak, diabaikan oleh S3 Lambda Objek.

Saat memanggil operasi API `WriteGetObjectResponse`, Amazon S3 membutuhkan rute dan meminta token dari konteks peristiwa. Untuk informasi selengkapnya, lihat Format konteks peristiwa dan penggunaan.

Parameter rute dan token permintaan diperlukan untuk menghubungkan respons `WriteGetObjectResult` dengan pemanggil asli. Meskipun selalu tepat untuk mencoba kembali respons 500 (Internal Server Error), karena token permintaan adalah token sekali pakai, upaya selanjutnya untuk menggunakannya mungkin menghasilkan respons kode status HTTP 400 (Bad Request). Meskipun panggilan ke `WriteGetObjectResponse` dengan rute dan permintaan token tidak perlu dibuat dari fungsi Lambda yang dipanggil, itu harus dibuat oleh identitas di akun yang sama. Panggilan itu juga harus diselesaikan sebelum fungsi Lambda selesai dieksekusi.

Bekerja dengan permintaan **`HeadObject`** di Lambda

Bagian ini mengasumsikan bahwa Titik Akses Lambda Objek Anda telah dikonfigurasi untuk memanggil fungsi Lambda `HeadObject`. Lambda akan menerima payload JSON yang berisi kunci bernama `headObjectContext`. Di dalam konteks, ada properti tunggal yang disebut `inputS3Url`, yang merupakan URL yang telah ditandatangani sebelumnya untuk titik akses pendukung untuk `HeadObject`.

URL yang telah ditandatangani sebelumnya akan menyertakan properti berikut jika ditentukan:

- `versionId` (di dalam parameter kueri)
- `requestPayer` (di dalam header `x-amz-request-payer`)
- `expectedBucketOwner` (di dalam header `x-amz-expected-bucket-owner`)

Properti lain tidak akan ditandatangani sebelumnya, dan karenanya tidak akan disertakan. Opsi yang tidak ditandatangani yang dikirim sebagai header dapat ditambahkan secara manual ke permintaan saat memanggil URL yang ditandatangani sebelumnya yang ditemukan di header `userRequest`. Opsi enkripsi di sisi server tidak didukung untuk `HeadObject`.

Untuk parameter URI sintaks permintaan, lihat `HeadObject` di Referensi API Amazon Simple Storage Service.

Contoh berikut ini menunjukkan muatan input Lambda JSON untuk `HeadObject`.

```
{
  "xAmzRequestId": "requestId",
  "**headObjectContext**": {
    "**inputS3Url**": "https://my-s3-ap-111122223333.s3-accesspoint.us-
east-1.amazonaws.com/example?X-Amz-Security-Token=<snip>"
  },
  "configuration": {
```

```
        "accessPointArn": "arn:aws:s3-object-lambda:us-
east-1:111122223333:accesspoint/example-object-lambda-ap",
        "supportingAccessPointArn": "arn:aws:s3:us-
east-1:111122223333:accesspoint/example-ap",
        "payload": "{}"
   },
   "userRequest": {
        "url": "https://object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com/example",
        "headers": {
            "Host": "object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com",
            "Accept-Encoding": "identity",
            "X-Amz-Content-SHA256": "e3b0c44298fc1example"
        }
    },
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "principalId",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/example",
        "accountId": "111122223333",
        "accessKeyId": "accessKeyId",
        "sessionContext": {
             "attributes": {
             "mfaAuthenticated": "false",
             "creationDate": "Wed Mar 10 23:41:52 UTC 2021"
        },
        "sessionIssuer": {
             "type": "Role",
             "principalId": "principalId",
             "arn": "arn:aws:iam::111122223333:role/Admin",
             "accountId": "111122223333",
             "userName": "Admin"
             }
        }
     },
   "protocolVersion": "1.00"
}
```

Fungsi Lambda Anda harus mengembalikan objek JSON yang berisi header dan nilai yang akan dikembalikan untuk panggilan `HeadObject`.

Contoh berikut ini menunjukkan struktur respons Lambda JSON untuk `HeadObject`.

```
{
    "statusCode": <number>; // Required
    "errorCode": <string>;
    "errorMessage": <string>;
    "headers": {
        "Accept-Ranges": <string>,
        "x-amz-archive-status": <string>,
        "x-amz-server-side-encryption-bucket-key-enabled": <boolean>,
        "Cache-Control": <string>,
        "Content-Disposition": <string>,
        "Content-Encoding": <string>,
        "Content-Language": <string>,
        "Content-Length": <number>, // Required
        "Content-Type": <string>,
        "x-amz-delete-marker": <boolean>,
        "ETag": <string>,
        "Expires": <string>,
        "x-amz-expiration": <string>,
        "Last-Modified": <string>,
        "x-amz-missing-meta": <number>,
        "x-amz-object-lock-mode": <string>,
        "x-amz-object-lock-legal-hold": <string>,
        "x-amz-object-lock-retain-until-date": <string>,
        "x-amz-mp-parts-count": <number>,
        "x-amz-replication-status": <string>,
        "x-amz-request-charged": <string>,
        "x-amz-restore": <string>,
        "x-amz-server-side-encryption": <string>,
        "x-amz-server-side-encryption-customer-algorithm": <string>,
        "x-amz-server-side-encryption-aws-kms-key-id": <string>,
        "x-amz-server-side-encryption-customer-key-MD5": <string>,
        "x-amz-storage-class": <string>,
        "x-amz-tagging-count": <number>,
        "x-amz-version-id": <string>,
        <x-amz-meta-headers>: <string>, // user-defined metadata
        "x-amz-meta-meta1": <string>, // example of the user-defined metadata header,
 it will need the x-amz-meta prefix
        "x-amz-meta-meta2": <string>
        ...
    };
}
```

Contoh berikut menunjukkan cara untuk menggunakan URL yang telah ditentukan sebelumnya untuk mengisi respons Anda dengan mengubah nilai header sesuai kebutuhan sebelum mengembalikan JSON.

Python

```
import requests

def lambda_handler(event, context):
    print(event)

    # Extract the presigned URL from the input.
    s3_url = event["headObjectContext"]["inputS3Url"]

    # Get the head of the object from S3.
    response = requests.head(s3_url)

    # Return the error to S3 Object Lambda (if applicable).
    if (response.status_code >= 400):
        return {
            "statusCode": response.status_code,
            "errorCode": "RequestFailure",
            "errorMessage": "Request to S3 failed"
    }

    # Store the headers in a dictionary.
    response_headers = dict(response.headers)

    # This obscures Content-Type in a transformation, it is optional to add
    response_headers["Content-Type"] = ""

    # Return the headers to S3 Object Lambda.
    return {
        "statusCode": response.status_code,
        "headers": response_headers
        }
```

Bekerja dengan permintaan **`ListObjects`** di Lambda

Bagian ini mengasumsikan bahwa Titik Akses Lambda Objek Anda telah dikonfigurasi untuk memanggil fungsi Lambda `ListObjects`. Lambda akan menerima payload JSON dengan

objek baru bernama `listObjectsContext`. `listObjectsContext`berisi properti tunggal, `inputS3Url`, yang merupakan URL yang telah ditandatangani sebelumnya untuk titik akses pendukung untuk `ListObjects`.

Berbeda dengan `GetObject` dan `HeadObject`, URL yang telah ditandatangani sebelumnya akan menyertakan properti berikut jika ditentukan:

- Semua parameter kueri
- `requestPayer` (di dalam header `x-amz-request-payer`)
- `expectedBucketOwner` (di dalam header `x-amz-expected-bucket-owner`)

Untuk parameter URI sintaks permintaan, lihat `ListObjects` di Referensi API Amazon Simple Storage Service.

> **Important**
>
> Kami menyarankan Anda menggunakan versi yang lebih baru, ListObjectsV2, saat mengembangkan aplikasi. Untuk kompatibilitas mundur, Amazon S3 terus mendukung `ListObjects`.

Contoh berikut ini menunjukkan muatan input Lambda JSON untuk `ListObjects`.

```
{
    "xAmzRequestId": "requestId",
     "**listObjectsContext**": {
     "**inputS3Url**": "https://my-s3-ap-111122223333.s3-accesspoint.us-
east-1.amazonaws.com/?X-Amz-Security-Token=<snip>",
     },
    "configuration": {
        "accessPointArn": "arn:aws:s3-object-lambda:us-
east-1:111122223333:accesspoint/example-object-lambda-ap",
        "supportingAccessPointArn": "arn:aws:s3:us-
east-1:111122223333:accesspoint/example-ap",
        "payload": "{}"
    },
    "userRequest": {
        "url": "https://object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com/example",
        "headers": {
```

```
            "Host": "object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com",
            "Accept-Encoding": "identity",
            "X-Amz-Content-SHA256": "e3b0c44298fc1example"
        }
    },
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "principalId",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/example",
        "accountId": "111122223333",
        "accessKeyId": "accessKeyId",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "Wed Mar 10 23:41:52 UTC 2021"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "principalId",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            }
        }
    },
  "protocolVersion": "1.00"
}
```

Fungsi Lambda Anda harus mengembalikan objek JSON yang berisi kode status, daftar hasil XML, atau informasi kesalahan yang akan dikembalikan dari S3 Lambda Objek.

S3 Lambda Objek tidak memproses atau memvalidasi `listResultXml`, melainkan meneruskannya ke pemanggil `ListObjects`. Untuk `listBucketResult`, S3 Lambda Objek mengharapkan properti tertentu memiliki jenis tertentu dan akan memunculkan pengecualian jika tidak dapat menguraikannya. `listResultXml` dan `listBucketResult` tidak dapat diberikan secara bersamaan.

Contoh berikut menunjukkan cara menggunakan URL yang telah ditandatangani untuk memanggil Amazon S3, dan menggunakan hasilnya untuk mengisi respons, termasuk pemeriksaan kesalahan.

Python

```
import requests
import xmltodict

def lambda_handler(event, context):
    # Extract the presigned URL from the input.
    s3_url = event["listObjectsContext"]["inputS3Url"]


    # Get the head of the object from Amazon S3.
    response = requests.get(s3_url)

    # Return the error to S3 Object Lambda (if applicable).
    if (response.status_code >= 400):
        error = xmltodict.parse(response.content)
        return {
            "statusCode": response.status_code,
            "errorCode": error["Error"]["Code"],
            "errorMessage": error["Error"]["Message"]
        }

    # Store the XML result in a dict.
    response_dict = xmltodict.parse(response.content)

    # This obscures StorageClass in a transformation, it is optional to add
    for item in response_dict['ListBucketResult']['Contents']:
        item['StorageClass'] = ""

    # Convert back to XML.
    listResultXml = xmltodict.unparse(response_dict)

    # Create response with listResultXml.
    response_with_list_result_xml = {
        'statusCode': 200,
        'listResultXml': listResultXml
    }

    # Create response with listBucketResult.
    response_dict['ListBucketResult'] =
 sanitize_response_dict(response_dict['ListBucketResult'])
    response_with_list_bucket_result = {
        'statusCode': 200,
        'listBucketResult': response_dict['ListBucketResult']
```

```
    }

    # Return the list to S3 Object Lambda.
    # Can return response_with_list_result_xml or response_with_list_bucket_result
    return response_with_list_result_xml

# Converting the response_dict's key to correct casing
def sanitize_response_dict(response_dict: dict):
    new_response_dict = dict()
    for key, value in response_dict.items():
        new_key = key[0].lower() + key[1:] if key != "ID" else 'id'
        if type(value) == list:
            newlist = []
            for element in value:
                if type(element) == type(dict()):
                    element = sanitize_response_dict(element)
                newlist.append(element)
            value = newlist
        elif type(value) == dict:
            value = sanitize_response_dict(value)
        new_response_dict[new_key] = value
    return new_response_dict
```

Contoh berikut ini menunjukkan struktur respons Lambda JSON untuk `ListObjects`.

```
{
  "statusCode": <number>; // Required
  "errorCode": <string>;
  "errorMessage": <string>;
  "listResultXml": <string>; // This can also be Error XML string in case S3 returned
 error response when calling the pre-signed URL

  "listBucketResult": {  // listBucketResult can be provided instead of listResultXml,
 however they can not both be provided in the JSON response
        "name": <string>,  // Required for 'listBucketResult'
        "prefix": <string>,
        "marker": <string>,
        "nextMarker": <string>,
        "maxKeys": <int>,   // Required for 'listBucketResult'
        "delimiter": <string>,
        "encodingType": <string>
        "isTruncated": <boolean>,  // Required for 'listBucketResult'
```

```
        "contents": [  {
            "key": <string>,  // Required for 'content'
            "lastModified": <string>,
            "eTag": <string>,
            "checksumAlgorithm": <string>,   // CRC32,  CRC32C,  SHA1,  SHA256
            "size": <int>,   // Required for 'content'
            "owner": {
                "displayName": <string>,  // Required for 'owner'
                "id": <string>,  // Required for 'owner'
            },
            "storageClass": <string>
            },
        ...
        ],
        "commonPrefixes": [  {
            "prefix": <string>   // Required for 'commonPrefix'
        },
        ...
        ],
    }
}
```

Bekerja dengan permintaan **`ListObjectsV2`** di Lambda

Bagian ini mengasumsikan bahwa Titik Akses Lambda Objek Anda telah dikonfigurasi untuk memanggil fungsi Lambda `ListObjectsV2`. Lambda akan menerima payload JSON dengan objek baru bernama `listObjectsV2Context`. `listObjectsV2Context`berisi properti tunggal, `inputS3Url`, yang merupakan URL yang telah ditandatangani sebelumnya untuk titik akses pendukung untuk `ListObjectsV2`.

Berbeda dengan `GetObject` dan `HeadObject`, URL yang telah ditandatangani sebelumnya akan menyertakan properti berikut jika ditentukan:

- Semua parameter kueri
- `requestPayer` (di dalam header `x-amz-request-payer`)
- `expectedBucketOwner` (di dalam header `x-amz-expected-bucket-owner`)

Untuk parameter URI sintaks permintaan, lihat `ListObjectsV2` di Referensi API Amazon Simple Storage Service.

Contoh berikut ini menunjukkan muatan input Lambda JSON untuk `ListObjectsV2`.

```
{
    "xAmzRequestId": "requestId",
     "**listObjectsV2Context**": {
     "**inputS3Url**": "https://my-s3-ap-111122223333.s3-accesspoint.us-
east-1.amazonaws.com/?list-type=2&X-Amz-Security-Token=<snip>",
     },
    "configuration": {
        "accessPointArn": "arn:aws:s3-object-lambda:us-
east-1:111122223333:accesspoint/example-object-lambda-ap",
        "supportingAccessPointArn": "arn:aws:s3:us-
east-1:111122223333:accesspoint/example-ap",
        "payload": "{}"
    },
    "userRequest": {
        "url": "https://object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com/example",
        "headers": {
            "Host": "object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com",
            "Accept-Encoding": "identity",
            "X-Amz-Content-SHA256": "e3b0c44298fc1example"
        }
    },
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "principalId",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/example",
        "accountId": "111122223333",
        "accessKeyId": "accessKeyId",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "Wed Mar 10 23:41:52 UTC 2021"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "principalId",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            }
        }
    },
```

```
  "protocolVersion": "1.00"
}
```

Fungsi Lambda Anda harus mengembalikan objek JSON yang berisi kode status, daftar hasil XML, atau informasi kesalahan yang akan dikembalikan dari S3 Lambda Objek.

S3 Lambda Objek tidak memproses atau memvalidasi `listResultXml`, melainkan meneruskannya ke pemanggil `ListObjectsV2`. Untuk `listBucketResult`, S3 Lambda Objek mengharapkan properti tertentu memiliki jenis tertentu dan akan memunculkan pengecualian jika tidak dapat menguraikannya. `listResultXml` dan `listBucketResult` tidak dapat diberikan secara bersamaan.

Contoh berikut menunjukkan cara menggunakan URL yang telah ditandatangani untuk memanggil Amazon S3, dan menggunakan hasilnya untuk mengisi respons, termasuk pemeriksaan kesalahan.

Python

```
import requests
import xmltodict

def lambda_handler(event, context):
    # Extract the presigned URL from the input.
    s3_url = event["listObjectsV2Context"]["inputS3Url"]


    # Get the head of the object from Amazon S3.
    response = requests.get(s3_url)

    # Return the error to S3 Object Lambda (if applicable).
    if (response.status_code >= 400):
        error = xmltodict.parse(response.content)
        return {
            "statusCode": response.status_code,
            "errorCode": error["Error"]["Code"],
            "errorMessage": error["Error"]["Message"]
        }

    # Store the XML result in a dict.
    response_dict = xmltodict.parse(response.content)

    # This obscures StorageClass in a transformation, it is optional to add
    for item in response_dict['ListBucketResult']['Contents']:
```

```
        item['StorageClass'] = ""

    # Convert back to XML.
    listResultXml = xmltodict.unparse(response_dict)

    # Create response with listResultXml.
    response_with_list_result_xml = {
        'statusCode': 200,
        'listResultXml': listResultXml
    }

    # Create response with listBucketResult.
    response_dict['ListBucketResult'] =
 sanitize_response_dict(response_dict['ListBucketResult'])
    response_with_list_bucket_result = {
        'statusCode': 200,
        'listBucketResult': response_dict['ListBucketResult']
    }

    # Return the list to S3 Object Lambda.
    # Can return response_with_list_result_xml or response_with_list_bucket_result
    return response_with_list_result_xml

# Converting the response_dict's key to correct casing
def sanitize_response_dict(response_dict: dict):
    new_response_dict = dict()
    for key, value in response_dict.items():
        new_key = key[0].lower() + key[1:] if key != "ID" else 'id'
        if type(value) == list:
            newlist = []
            for element in value:
                if type(element) == type(dict()):
                    element = sanitize_response_dict(element)
                newlist.append(element)
            value = newlist
        elif type(value) == dict:
            value = sanitize_response_dict(value)
        new_response_dict[new_key] = value
    return new_response_dict
```

Contoh berikut ini menunjukkan struktur respons Lambda JSON untuk `ListObjectsV2`.

```
{
    "statusCode": <number>; // Required
    "errorCode": <string>;
    "errorMessage": <string>;
    "listResultXml": <string>; // This can also be Error XML string in case S3 returned
 error response when calling the pre-signed URL

    "listBucketResult": {  // listBucketResult can be provided instead of
 listResultXml, however they can not both be provided in the JSON response
        "name": <string>, // Required for 'listBucketResult'
        "prefix": <string>,
        "startAfter": <string>,
        "continuationToken": <string>,
        "nextContinuationToken": <string>,
        "keyCount": <int>, // Required for 'listBucketResult'
        "maxKeys": <int>, // Required for 'listBucketResult'
        "delimiter": <string>,
        "encodingType": <string>
        "isTruncated": <boolean>, // Required for 'listBucketResult'
        "contents": [ {
            "key": <string>, // Required for 'content'
            "lastModified": <string>,
            "eTag": <string>,
            "checksumAlgorithm": <string>, // CRC32, CRC32C, SHA1, SHA256
            "size": <int>, // Required for 'content'
            "owner": {
                "displayName": <string>, // Required for 'owner'
                "id": <string>, // Required for 'owner'
            },
            "storageClass": <string>
            },
            ...
        ],
        "commonPrefixes": [ {
            "prefix": <string> // Required for 'commonPrefix'
            },
        ...
        ],
    }
}
```

Format konteks peristiwa dan penggunaan

Amazon S3 Object Lambda menyediakan konteks tentang permintaan yang sedang dibuat jika diteruskan ke fungsi Anda. AWS Lambda Berikut adalah contoh permintaannya. Deskripsi bidang disertakan setelah contoh.

```
{
    "xAmzRequestId": "requestId",
    "getObjectContext": {
        "inputS3Url": "https://my-s3-ap-111122223333.s3-accesspoint.us-
east-1.amazonaws.com/example?X-Amz-Security-Token=<snip>",
        "outputRoute": "io-use1-001",
        "outputToken": "OutputToken"
    },
    "configuration": {
        "accessPointArn": "arn:aws:s3-object-lambda:us-
east-1:111122223333:accesspoint/example-object-lambda-ap",
        "supportingAccessPointArn": "arn:aws:s3:us-
east-1:111122223333:accesspoint/example-ap",
        "payload": "{}"
    },
    "userRequest": {
        "url": "https://object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com/example",
        "headers": {
            "Host": "object-lambda-111122223333.s3-object-lambda.us-
east-1.amazonaws.com",
            "Accept-Encoding": "identity",
            "X-Amz-Content-SHA256": "e3b0c44298fc1example"
        }
    },
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "principalId",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/example",
        "accountId": "111122223333",
        "accessKeyId": "accessKeyId",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "Wed Mar 10 23:41:52 UTC 2021"
            },
            "sessionIssuer": {
```

```
                "type": "Role",
                "principalId": "principalId",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            }
        }
    },
    "protocolVersion": "1.00"
}
```

Kolom permintaan disertakan dalam permintaan berikut:

- `xAmzRequestId`–ID permintaan Amazon S3 untuk permintaan ini. Kami merekomendasikan agar Anda mencatat nilai ini untuk membantu dengan debugging.
- `getObjectContext`–Detail input dan output untuk koneksi ke Amazon S3 dan S3 Lambda Objek.
  - `inputS3Url`–URL yang telah ditandatangani sebelumnya, yang dapat digunakan untuk mengambil objek asli dari Amazon S3. URL ditandatangani dengan menggunakan identitas pemanggil asli, dan izin pengguna tersebut akan berlaku saat URL tersebut digunakan. Jika ada header ditandatangani di URL, fungsi Lambda harus menyertakan header ini dalam panggilan ke Amazon S3, kecuali untuk header `Host`.
  - `outputRoute`–Token perutean yang ditambahkan ke URL S3 Lambda Objek ketika fungsi Lambda memanggil `WriteGetObjectResponse`.
  - `outputToken`–Token buram yang digunakan oleh S3 Lambda Objek untuk mencocokkan panggilan `WriteGetObjectResponse` dengan pemanggil asli.
- `configuration`–Informasi konfigurasi tentang Titik Akses Lambda Objek.
  - `accessPointArn`–Amazon Resource Name (ARN) dari Titik Akses Lambda Objek yang menerima permintaan ini.
  - `supportingAccessPointArn`–ARN dari titik akses pendukung yang ditentukan dalam konfigurasi Titik Akses Lambda Objek.
  - `payload`–Data kustom yang diterapkan pada konfigurasi Titik Akses Lambda Objek. S3 Lambda Objek memperlakukan data ini sebagai string buram, sehingga mungkin perlu diterjemahkan sebelum digunakan.
- `userRequest`–Informasi tentang panggilan asli untuk S3 Lambda Objek.
  - `url`–URL permintaan yang didekodekan seperti yang diterima oleh S3 Lambda Objek, tidak termasuk parameter kueri terkait otorisasi.

  - `headers`–Peta string ke string yang berisi header HTTP serta nilai dari panggilan aslinya, tidak termasuk header terkait otorisasi. Jika header yang sama muncul beberapa kali, nilai dari setiap instans header yang sama digabungkan ke dalam daftar yang dibatasi koma. Kasus header asli dipertahankan dalam peta ini.
- `userIdentity`–Detail tentang identitas yang melakukan panggilan ke S3 Lambda Objek. Untuk informasi lebih lanjut, lihat Peristiwa Pencatatan Data untuk Pelacakan dalam AWS CloudTrail Panduan Pengguna.
  - `type`–Jenis identitas.
  - `accountId`— Akun AWS yang menjadi milik identitas.
  - `userName`–Nama ramah dari identitas yang melakukan panggilan.
  - `principalId`–Pengidentifikasi unik untuk identitas yang melakukan panggilan.
  - `arn`–ARN pengguna utama yang melakukan panggilan. Bagian terakhir ARN berisi pengguna atau peran yang melakukan panggilan.
  - `sessionContext`–Jika permintaan dibuat dengan kredensial keamanan sementara, elemen ini memberikan informasi tentang sesi yang dibuat untuk kredensial tersebut.
  - `invokedBy`— Nama Layanan AWS yang membuat permintaan, seperti Amazon EC2 Auto Scaling atau. AWS Elastic Beanstalk
  - `sessionIssuer`–Jika permintaan dibuat dengan kredensial keamanan sementara, elemen ini memberikan informasi tentang bagaimana kredensial tersebut diperoleh.
- `protocolVersion`–ID Versi dari konteks yang disediakan. Format bidang ini adalah `{Major Version}.{Minor Version}`. Nomor versi minor selalu berupa angka dua digit. Penghapusan atau perubahan apa pun pada semantik suatu bidang memerlukan perubahan versi utama dan memerlukan keikutsertaan aktif. Amazon S3 dapat menambahkan kolom baru kapan saja, sehingga Anda mungkin mengalami peningkatan versi kecil. Karena sifat peluncuran perangkat lunak, Anda mungkin melihat beberapa versi minor digunakan sekaligus.

Bekerja dengan header Range dan partNumber

Saat bekerja dengan objek besar di Lambda Objek Amazon S3, Anda dapat menggunakan header HTTP `Range` untuk mengunduh rentang byte tertentu dari sebuah objek. Untuk mengambil rentang byte berbeda dari dalam objek yang sama, Anda dapat menggunakan koneksi bersamaan ke Amazon S3. Anda juga dapat menentukan parameter `partNumber` (integer antara 1 dan 10.000), yang melakukan permintaan berkisar untuk bagian objek tertentu.

Karena ada beberapa cara yang mungkin Anda inginkan untuk menangani permintaan yang menyertakan parameter `Range` atau `partNumber`, S3 Lambda Objek tidak menerapkan parameter ini ke objek yang ditransformasi. Sebagai gantinya, AWS Lambda fungsi Anda harus mengimplementasikan fungsi ini sesuai kebutuhan untuk aplikasi Anda.

Untuk menggunakan parameter `Range` dan `partNumber` dengan S3 Lambda Objek, Anda melakukan hal berikut ini:

- Aktifkan parameter ini di konfigurasi Titik Akses Lambda Objek.
- Tulis fungsi Lambda yang dapat menangani permintaan yang menyertakan parameter ini.

Langkah-langkah berikut menjelaskan cara untuk mencapai hal ini.

Langkah 1: Mengonfigurasi Titik Akses Lambda Objek Anda

Secara default, Titik Akses Objek Lambda merespons dengan kesalahan kode status HTTP 501 (Tidak Diimplementasikan) terhadap permintaan `GetObject` atau `HeadObject` yang berisi parameter `Range` atau `partNumber`, baik di header atau parameter kueri.

Untuk mengaktifkan Titik Akses Lambda Objek untuk menerima permintaan tersebut, Anda harus menyertakan `GetObject-Range`, `GetObject-PartNumber`, `HeadObject-Range`, atau `HeadObject-PartNumber` dalam bagian `AllowedFeatures` konfigurasi Titik Akses Lambda Objek Anda. Untuk informasi selengkapnya tentang memperbarui konfigurasi Titik Akses Lambda Objek, lihat Membuat Titik Akses Objek Lambda.

Langkah 2: Menerapkan penanganan **`Range`** atau **`partNumber`** di dalam fungsi Lambda Anda

Saat Titik Akses Lambda Object Anda menginvokasi fungsi Lambda Anda dengan rentang permintaan `GetObject` atau `HeadObject`, parameter `Range` atau `partNumber` disertakan dalam konteks peristiwa. Lokasi parameter dalam konteks peristiwa tergantung pada parameter mana yang digunakan dan bagaimana itu dimasukkan dalam permintaan asli ke Titik Akses Lambda Object, seperti yang dijelaskan dalam tabel berikut.

Parameter	Lokasi konteks peristiwa
`Range` (header)	`userRequest.headers.Range`
`Range` (parameter kueri)	`userRequest.url` (parameter kueri `Range`)

Parameter	Lokasi konteks peristiwa
`partNumber`	`userRequest.url` (parameter kueri `partNumber`)

> **Important**
>
> URL presigned yang disediakan untuk Titik Akses Lambda Objek Anda tidak berisi parameter `Range` atau `partNumber` dari permintaan asli. Lihat opsi berikut tentang cara menangani parameter ini dalam AWS Lambda fungsi Anda.

Setelah Anda mengekstrak nilai `Range` atau `partNumber`, Anda dapat mengambil salah satu pendekatan berikut ini, berdasarkan kebutuhan aplikasi Anda:

A. Petakan **`Range`** yang diminta atau **`partNumber`** ke objek yang diubah (disarankan).

Cara yang paling dapat diandalkan untuk menangani permintaan `Range` atau `partNumber` adalah dengan melakukan hal berikut ini:

- Mengambil objek lengkap dari Amazon S3.
- Mengubah objek tersebut.
- Terapkan parameter `Range` atau `partNumber` yang diminta ke objek yang diubah.

Untuk melakukan ini, gunakan URL yang telah ditandatangani sebelumnya untuk mengambil seluruh objek dari Amazon S3, dan kemudian memproses objek tersebut sesuai kebutuhan. Untuk contoh fungsi Lambda yang memproses `Range` parameter dengan cara ini, lihat sampel ini di repositori AWS Sampel GitHub .

B. Petakan **`Range`** yang diminta ke URL yang telah ditandatangani sebelumnya.

Dalam beberapa kasus, fungsi Lambda Anda dapat memetakan permintaan `Range` langsung ke URL yang telah ditentukan sebelumnya untuk mengambil hanya sebagian objek dari Amazon S3. Pendekatan ini hanya sesuai jika transformasi Anda memenuhi kedua kriteria berikut:

1. Fungsi transformasi Anda dapat diterapkan ke sebagian rentang objek.
2. Menerapkan parameter `Range` sebelum atau sesudah fungsi transformasi menghasilkan objek yang ditransformasikan yang sama.

Misalnya, fungsi transformasi yang mengubah semua karakter dalam objek berkode ASCII menjadi huruf besar memenuhi kedua kriteria sebelumnya. Transformasi dapat diterapkan pada bagian dari suatu objek, dan menerapkan parameter `Range` sebelum transformasi mencapai hasil yang sama seperti menerapkannya setelah transformasi.

Sebaliknya, fungsi yang membalikkan karakter dalam objek yang dikodekan ASCII tidak memenuhi kriteria ini. Fungsi seperti itu memenuhi kriteria 1, karena dapat diterapkan pada rentang objek sebagian. Namun, itu tidak memenuhi kriteria 2, karena menerapkan parameter `Range` sebelum transformasi mencapai hasil yang berbeda daripada menerapkan parameter setelah transformasi.

Pertimbangkan permintaan untuk menerapkan fungsi ke tiga karakter pertama dari suatu objek dengan isinya `abcdefg`. Menerapkan parameter `Range` sebelum transformasi hanya mengambil `abc` dan kemudian membalikkan data, mengembalikan `cba`. Tetapi jika parameter diterapkan setelah transformasi, fungsi mengambil seluruh objek, membalikkannya, lalu menerapkan parameter `Range`, mengembalikan `gfe`. Karena hasil ini berbeda, fungsi ini tidak boleh menerapkan parameter `Range` saat mengambil objek dari Amazon S3. Sebaliknya, itu harus mengambil seluruh objek, melakukan transformasi, dan hanya kemudian menerapkan parameter `Range`.

> **Warning**
>
> Dalam banyak kasus, menerapkan parameter `Range` ke URL yang telah ditandatangani sebelumnya akan menghasilkan perilaku tak terduga oleh fungsi Lambda atau klien yang meminta. Kecuali Anda yakin bahwa aplikasi Anda akan berfungsi dengan baik ketika hanya mengambil sebagian objek dari Amazon S3, kami menyarankan Anda mengambil dan mengubah objek penuh seperti yang dijelaskan sebelumnya dalam pendekatan A.

Jika aplikasi Anda memenuhi kriteria yang dijelaskan sebelumnya dalam pendekatan B, Anda dapat menyederhanakan AWS Lambda fungsi Anda dengan mengambil hanya rentang objek yang diminta dan kemudian menjalankan transformasi Anda pada rentang tersebut.

Contoh kode Java berikut mendemonstrasikan cara untuk melakukan hal berikut ini:

- Ambil header `Range` dari permintaan `GetObject`.
- Tambahkan header `Range` ke URL yang telah ditandatangani sebelumnya, yang dapat digunakan oleh Lambda untuk mengambil rentang yang diminta dari Amazon S3.

```
private HttpRequest.Builder applyRangeHeader(ObjectLambdaEvent event,
 HttpRequest.Builder presignedRequest) {
    var header = event.getUserRequest().getHeaders().entrySet().stream()
            .filter(e -> e.getKey().toLowerCase(Locale.ROOT).equals("range"))
            .findFirst();

    // Add check in the query string itself.
    header.ifPresent(entry -> presignedRequest.header(entry.getKey(),
 entry.getValue()));
    return presignedRequest;
}
```

Menggunakan fungsi Lambda yang AWS dibangun

AWS menyediakan beberapa AWS Lambda fungsi bawaan yang dapat Anda gunakan dengan Amazon S3 Object Lambda untuk mendeteksi dan menyunting informasi identitas pribadi (PII) dan mendekompresi objek S3. Fungsi Lambda ini tersedia di AWS Serverless Application Repository. Anda dapat memilih fungsi-fungsi ini melalui AWS Management Console ketika Anda membuat Titik Akses Lambda Objek Anda.

Untuk informasi selengkapnya tentang cara menerapkan aplikasi tanpa server dari aplikasi AWS Serverless Application Repository, lihat Menerapkan Aplikasi di Panduan Pengembang.AWS Serverless Application Repository

> **Note**
>
> Contoh berikut hanya dapat digunakan dengan permintaan `GetObject`.

Contoh 1: Kontrol akses PII

Fungsi Lambda ini menggunakan Amazon Comprehend, layanan pemrosesan bahasa alami (NLP) yang menggunakan machine learning untuk menemukan wawasan dan hubungan dalam teks. Fungsi ini secara otomatis mendeteksi informasi pengenal pribadi (PII), seperti nama, alamat, tanggal, nomor kartu kredit, dan nomor jaminan sosial dalam dokumen di bucket Amazon S3 Anda. Jika Anda memiliki dokumen di bucket Anda yang menyertakan PII, Anda dapat mengonfigurasi fungsi Kontrol Akses PII untuk mendeteksi jenis entitas PII ini dan membatasi akses ke pengguna yang tidak sah.

Untuk memulai, terapkan fungsi Lambda berikut di akun Anda dan tambahkan Amazon Resource Name (ARN) untuk fungsi tersebut ke konfigurasi Titik Akses Lambda Objek Anda.

Berikut adalah contoh ARN untuk fungsi ini:

```
arn:aws:serverlessrepo:us-east-1:111122223333:applications/
ComprehendPiiAccessControlS3ObjectLambda
```

Anda dapat menambahkan atau melihat fungsi ini pada AWS Management Console dengan menggunakan AWS Serverless Application Repository link berikut: ComprehendPiiAccessControlS3 ObjectLambda.

Untuk melihat fungsi ini aktif GitHub, lihat Amazon Comprehend S3 Object Lambda.

Contoh 2: Penyamaran PII

Fungsi Lambda ini menggunakan Amazon Comprehend, layanan pemrosesan bahasa alami (NLP) yang menggunakan machine learning untuk menemukan wawasan dan hubungan dalam teks. Fungsi ini secara otomatis menyunting informasi pengenal pribadi (PII), seperti nama, alamat, tanggal, nomor kartu kredit, dan nomor jaminan sosial dari dokumen di bucket Amazon S3 Anda.

Jika Anda memiliki dokumen di bucket Anda yang menyertakan informasi seperti nomor kartu kredit atau informasi rekening bank, Anda dapat mengonfigurasi fungsi Lambda PII Redaction S3 Object untuk mendeteksi PII dan kemudian mengembalikan salinan dokumen ini di mana jenis entitas PII disunting.

Untuk memulai, deploy fungsi Lambda berikut ini di akun Anda, dan tambahkan ARN untuk fungsi tersebut ke konfigurasi Titik Akses Lambda Objek Anda.

Berikut adalah contoh ARN untuk fungsi ini:

```
arn:aws:serverlessrepo:us-east-1:111122223333::applications/
ComprehendPiiRedactionS3ObjectLambda
```

Anda dapat menambahkan atau melihat fungsi ini pada AWS Management Console dengan menggunakan AWS Serverless Application Repository link berikut: ComprehendPiiRedactionS3 ObjectLambda.

Untuk melihat fungsi ini aktif GitHub, lihat Amazon Comprehend S3 Object Lambda.

Untuk mempelajari tentang end-to-end prosedur lengkap untuk beberapa tugas Lambda Objek S3 dalam redaksi PII, lihat. Tutorial: Mendeteksi dan menyunting data PII dengan S3 Object Lambda dan Amazon Comprehend

Contoh 3: Dekompresi

Fungsi Lambda `S3ObjectLambdaDecompression` dapat mendekompresi objek yang disimpan di Amazon S3 dalam salah satu dari enam format file terkompresi: `bzip2`, `gzip`, `snappy`, `zlib`, `zstandard`, dan `ZIP`.

Untuk memulai, deploy fungsi Lambda berikut ini di akun Anda, dan tambahkan ARN untuk fungsi tersebut ke konfigurasi Titik Akses Lambda Objek Anda.

Berikut adalah contoh ARN untuk fungsi ini:

```
arn:aws:serverlessrepo:us-east-1:111122223333::applications/S3ObjectLambdaDecompression
```

Anda dapat menambahkan atau melihat fungsi ini pada AWS Management Console dengan menggunakan AWS Serverless Application Repository link berikut: S3 ObjectLambdaDecompression.

Untuk melihat fungsi ini aktif GitHub, lihat Dekompresi Lambda Objek S3.

Praktik terbaik dan pedoman untuk S3 Lambda Objek

Saat menggunakan S3 Lambda Objek, ikuti praktik terbaik dan pedoman ini untuk mengoptimalkan operasi dan performanya.

Topik

- Bekerja dengan S3 Lambda Objek
- Layanan AWS digunakan sehubungan dengan S3 Object Lambda
- Header Range dan partNumber
- Mengubah expiry-date
- Bekerja dengan AWS CLI dan AWS SDK

Bekerja dengan S3 Lambda Objek

S3 Lambda Objek hanya mendukung pemrosesan permintaan `GET`, `LIST`, dan `HEAD`. Permintaan lain apa pun tidak memanggil AWS Lambda dan sebagai gantinya mengembalikan respons API standar yang tidak diubah. Anda dapat membuat maksimum 1.000 Titik Akses Objek Lambda Akun AWS

per Wilayah. AWS Lambda Fungsi yang Anda gunakan harus sama Akun AWS dan Region sebagai Object Lambda Access Point.

S3 Lambda Objek memungkinkan hingga 60 detik untuk mengalirkan respons lengkap ke pemanggilnya. Fungsi Anda juga tunduk pada kuota AWS Lambda default. Untuk informasi selengkapnya, lihat Kuota Lambda dalam Panduan Pengembang AWS Lambda .

Saat S3 Lambda Objek menginvokasi fungsi Lambda yang Anda tentukan, Anda bertanggung jawab untuk memastikan bahwa data apa pun yang ditimpa atau dihapus dari Amazon S3 oleh fungsi atau aplikasi Lambda yang Anda tentukan memang tepat dan benar.

Anda hanya dapat menggunakan S3 Lambda Objek untuk melakukan operasi pada objek. Anda tidak dapat menggunakan Lambda Objek S3 untuk melakukan operasi Amazon S3 lainnya, seperti memodifikasi atau menghapus bucket. Untuk mengetahui daftar lengkap pengoperasian S3 yang mendukung titik akses, lihat Kompatibilitas titik akses dengan operasi S3.

Selain daftar ini, Titik Akses Lambda Objek tidak mendukung `POST Object`, `CopyObject` (sebagai sumber), dan operasi API `SelectObjectContent`.

Layanan AWS digunakan sehubungan dengan S3 Object Lambda

S3 Object Lambda menghubungkan Amazon S3 AWS Lambda, dan secara opsional, pilihan Layanan AWS lain yang Anda pilih untuk mengirimkan objek yang relevan dengan aplikasi yang meminta. Semua yang Layanan AWS digunakan dengan S3 Object Lambda diatur oleh Perjanjian Tingkat Layanan (SLA) masing-masing. Misalnya, jika ada yang Layanan AWS tidak memenuhi Komitmen Layanannya, Anda berhak menerima Kredit Layanan, sebagaimana didokumentasikan dalam SLA layanan.

Header **`Range`** dan **`partNumber`**

Saat bekerja dengan objek besar, Anda dapat menggunakan header HTTP `Range` untuk mengunduh rentang byte tertentu dari suatu objek. Saat Anda menggunakan header `Range`, permintaan Anda hanya mengambil bagian tertentu dari objek. Anda juga dapat menggunakan header `partNumber` untuk melakukan permintaan berkisar untuk bagian tertentu dari objek.

Untuk informasi selengkapnya, lihat Bekerja dengan header Range dan partNumber.

Mengubah **`expiry-date`**

Anda dapat membuka atau mengunduh objek yang diubah dari Titik Akses Objek Lambda Anda di file. AWS Management Console Benda-benda ini harus berupa non-kedaluwarsa. Jika fungsi Lambda

Anda mengubah `expiry-date` dari objek Anda, Anda mungkin melihat objek kedaluwarsa yang tidak dapat dibuka atau diunduh. Perilaku ini hanya berlaku untuk objek yang dipulihkan S3 Glacier Fleksibel Retrieval dan S3 Glacier Deep Archive.

Bekerja dengan AWS CLI dan AWS SDK

AWS Command Line Interface (AWS CLI) Subperintah S3 (`cp`,`mv`, dan`sync`) dan penggunaan AWS SDK for Java `TransferManager` kelas tidak didukung untuk digunakan dengan S3 Object Lambda.

Tutorial S3 Lambda Objek

Tutorial berikut menyajikan end-to-end prosedur lengkap untuk beberapa tugas Lambda Objek S3.

- Tutorial: Mengubah data untuk aplikasi Anda dengan S3 Object Lambda
- Tutorial: Mendeteksi dan menyunting data PII dengan S3 Object Lambda dan Amazon Comprehend
- Tutorial: Menggunakan S3 Lambda Objek untuk menandai gambar secara dinamis saat gambarnya diambil

Men-debug S3 Lambda Objek

Permintaan ke titik akses Lambda Objek Amazon S3 mungkin menghasilkan respons kesalahan baru ketika terjadi kesalahan dengan invokasi atau eksekusi fungsi Lambda. Kesalahan ini mengikuti format yang sama dengan kesalahan standar Amazon S3. Untuk informasi tentang kesalahan S3 Lambda Objek, lihat Daftar Kode Kesalahan S3 Lambda Objek DALAM Referensi API Amazon Simple Storage Service.

Untuk informasi selengkapnya tentang fungsi Lambda umum debugging, lihat Pemantauan dan pemecahan masalah aplikasi DALAM Panduan Pengembang AWS Lambda .

Untuk informasi tentang kesalahan Amazon S3 standar, lihat Respons kesalahan dalam Referensi API Layanan Amazon Simple Storage Service.

Anda dapat mengaktifkan metrik permintaan di Amazon CloudWatch untuk Titik Akses Objek Lambda Anda. Metrik ini membantu Anda untuk memantau performa operasional titik akses Anda. Anda dapat mengaktifkan metrik permintaan selama, atau setelah pembuatan Titik Akses Lambda Objek. Untuk informasi selengkapnya, lihat Metrik permintaan Lambda Objek S3 di CloudWatch.

Untuk mendapatkan pencatatan log yang lebih terperinci tentang permintaan yang dibuat ke Titik Akses Lambda Objek Anda, Anda dapat mengaktifkan AWS CloudTrail peristiwa data. Untuk informasi lebih lanjut, lihat Peristiwa Pencatatan Data untuk Pelacakan dalam AWS CloudTrail Panduan Pengguna.

Untuk tutorial S3 Lambda Objek, lihat hal berikut ini:

- Tutorial: Mengubah data untuk aplikasi Anda dengan S3 Object Lambda
- Tutorial: Mendeteksi dan menyunting data PII dengan S3 Object Lambda dan Amazon Comprehend
- Tutorial: Menggunakan S3 Lambda Objek untuk menandai gambar secara dinamis saat gambarnya diambil

Untuk informasi lebih lanjut tentang titik akses standar, lihat Mengelola akses data dengan titik akses Amazon S3.

Untuk informasi tentang bekerja dengan bucket, lihat Gambaran umum bucket. Untuk informasi tentang bekerja dengan objek, lihat Gambaran umum objek Amazon S3.

Apa itu S3 Express One Zone?

Amazon S3 Express One Zone adalah kelas penyimpanan Amazon S3 tunggal berkinerja tinggi yang dibuat khusus untuk menghadirkan akses data milidetik satu digit yang konsisten untuk aplikasi Anda yang paling sensitif terhadap latensi. S3 Express One Zone adalah kelas penyimpanan cloud-object latensi terendah yang tersedia saat ini, dengan kecepatan akses data hingga 10x lebih cepat dan dengan biaya permintaan 50 persen lebih rendah dari Standar S3. Aplikasi dapat memperoleh manfaat segera dari permintaan yang diselesaikan hingga urutan besarnya lebih cepat. S3 Express One Zone memberikan elastisitas kinerja yang serupa dengan kelas penyimpanan S3 lainnya.

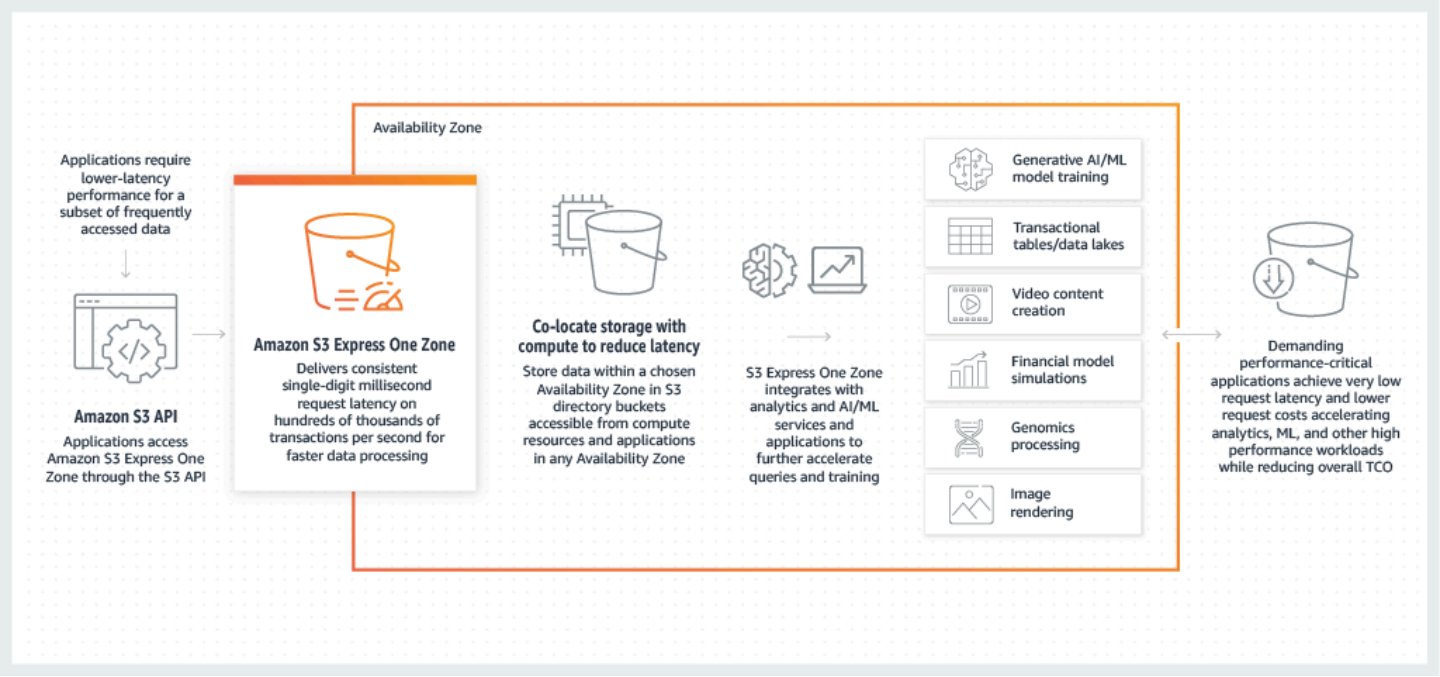
Seperti kelas penyimpanan Amazon S3 lainnya, Anda tidak perlu merencanakan atau menyediakan persyaratan kapasitas atau throughput terlebih dahulu. Anda dapat menskalakan penyimpanan Anda ke atas atau ke bawah, berdasarkan kebutuhan, dan mengakses data Anda melalui Amazon S3 API.

S3 Express One Zone adalah kelas penyimpanan S3 pertama di mana Anda dapat memilih satu Zona Ketersediaan dengan opsi untuk menempatkan bersama penyimpanan objek dengan sumber daya komputasi Anda, yang memberikan kecepatan akses setinggi mungkin. Selain itu, untuk lebih meningkatkan kecepatan akses dan mendukung ratusan ribu permintaan per detik, data di kelas penyimpanan S3 Express One Zone disimpan dalam tipe bucket baru: bucket direktori Amazon S3. Setiap bucket direktori dapat mendukung ratusan ribu transaksi per detik (TPS), tanpa memandang nama kunci atau pola aksesnya.

Kelas penyimpanan Amazon S3 Express One Zone dirancang untuk ketersediaan 99,95 persen dalam satu Availability Zone dan didukung oleh Perjanjian Tingkat Layanan Amazon S3. Dengan S3 Express One Zone, data Anda disimpan secara berlebihan di beberapa perangkat dalam satu Zona Ketersediaan. S3 Express One Zone dirancang untuk menangani kegagalan perangkat bersamaan dengan cepat mendeteksi dan memperbaiki setiap redundansi yang hilang. Apabila perangkat yang ada mengalami kegagalan, S3 Express One Zone secara otomatis mengalihkan permintaan ke perangkat baru dalam Zona Ketersediaan. Redundansi ini membantu memastikan akses tanpa gangguan ke data Anda dalam Zona Ketersediaan.

S3 Express One Zone sangat ideal untuk aplikasi apa pun yang penting untuk meminimalkan latensi yang diperlukan untuk mengakses objek. Aplikasi semacam itu dapat berupa alur kerja interaktif manusia, seperti pengeditan video, di mana para profesional kreatif membutuhkan akses responsif ke konten dari antarmuka pengguna mereka. S3 Express One Zone juga menguntungkan beban kerja analitik dan machine learning yang memiliki persyaratan responsif serupa dari data mereka, terutama beban kerja dengan banyak akses yang lebih kecil atau sejumlah besar akses acak. S3 Express One

Zone dapat digunakan bersama yang lain Layanan AWS untuk mendukung beban kerja analitik dan kecerdasan buatan dan pembelajaran mesin (AI/ML), seperti Amazon EMR, Amazon, SageMaker dan Amazon Athena.



Saat menggunakan S3 Express One Zone, Anda dapat berinteraksi dengan bucket direktori Anda di cloud pribadi virtual (VPC) dengan menggunakan titik akhir VPC gateway. Dengan titik akhir gateway, Anda dapat mengakses bucket direktori S3 Express One Zone dari VPC Anda tanpa gateway internet atau perangkat NAT untuk VPC Anda, dan tanpa biaya tambahan.

Anda dapat menggunakan banyak operasi dan fitur API Amazon S3 yang sama dengan bucket direktori yang Anda gunakan dengan bucket tujuan umum dan kelas penyimpanan lainnya. Ini termasuk Mountpoint untuk Amazon S3, enkripsi di sisi server dengan kunci yang dikelola Amazon S3 (SSE-S3), Operasi Batch S3, dan Blokir Akses Publik S3. Anda dapat mengakses S3 Express One Zone dengan menggunakan konsol Amazon S3 AWS Command Line Interface ,AWS CLI(), SDK AWS , dan Amazon S3 REST API.

Untuk informasi selengkapnya tentang S3 Express One Zone, lihat topik berikut.

- Gambaran Umum
- Fitur S3 Express One Zone
- Layanan terkait
- Langkah selanjutnya

Gambaran Umum

Untuk mengoptimalkan kinerja dan mengurangi latensi, S3 Express One Zone memperkenalkan konsep baru berikut.

Zona Ketersediaan Tunggal

Kelas penyimpanan Amazon S3 Express One Zone dirancang untuk ketersediaan 99,95 persen dalam satu Availability Zone dan didukung oleh Perjanjian Tingkat Layanan Amazon S3. Dengan S3 Express One Zone, data Anda disimpan secara berlebihan di beberapa perangkat dalam satu Zona Ketersediaan. S3 Express One Zone dirancang untuk menangani kegagalan perangkat bersamaan dengan cepat mendeteksi dan memperbaiki setiap redundansi yang hilang. Apabila perangkat yang ada mengalami kegagalan, S3 Express One Zone secara otomatis mengalihkan permintaan ke perangkat baru dalam Zona Ketersediaan. Redundansi ini membantu memastikan akses tanpa gangguan ke data Anda dalam Zona Ketersediaan.

Zona Ketersediaan adalah satu atau lebih pusat data diskret dengan daya redundan, jaringan, dan konektivitas dalam Wilayah AWS. Saat membuat bucket direktori, Anda memilih Availability Zone dan Wilayah AWS lokasi bucket Anda.

Bucket direktori

Ada dua jenis bucket Amazon S3: ember tujuan umum S3 dan ember direktori S3. Bucket tujuan umum adalah tipe bucket Amazon S3 default yang digunakan untuk sebagian besar kasus penggunaan S3. Bucket direktori hanya menggunakan kelas penyimpanan S3 Express One Zone, yang dirancang untuk beban kerja atau aplikasi yang kritis kinerja yang memerlukan latensi milidetik satu digit yang konsisten. Pilih jenis bucket yang paling sesuai dengan aplikasi dan persyaratan kinerja Anda.

Bucket direktori mengatur data secara hierarkis ke dalam direktori, sebagai lawan dari struktur penyimpanan datar dari bucket tujuan umum. Tidak ada batas prefiks untuk bucket direktori dan direktori individu dapat menskalakan secara horizontal.

Bucket direktori menggunakan kelas penyimpanan S3 Express One Zone, yang dibuat untuk digunakan oleh aplikasi yang sensitif terhadap kinerja. Dengan S3 Express One Zone, Anda dapat memilih satu Availability Zone dengan opsi untuk menempatkan penyimpanan objek bersama dengan sumber daya komputasi Anda, yang memberikan kecepatan akses setinggi mungkin. Ini tidak seperti bucket tujuan umum, yang secara berlebihan menyimpan objek di beberapa Availability Zone di. Wilayah AWS

Untuk informasi selengkapnya tentang bucket direktori, lihat Ember direktori. Untuk informasi selengkapnya tentang bucket tujuan umum, lihat Gambaran umum bucket.

Titik akhir dan titik akhir VPC gateway

Operasi API manajemen ember untuk bucket direktori tersedia melalui titik akhir Regional dan disebut sebagai operasi API titik akhir Regional. Contoh operasi API titik akhir Regional adalah `CreateBucket` dan `DeleteBucket`. Setelah membuat bucket direktori, Anda dapat menggunakan operasi API titik akhir Zonal untuk mengunggah dan mengelola objek di bucket direktori. Operasi API titik akhir Zona tersedia melalui titik akhir Zona. Contoh operasi API titik akhir Zonal adalah `PutObject` dan `CopyObject`.

Anda dapat mengakses S3 Express One Zone dari VPC Anda dengan menggunakan titik akhir VPC gateway. Setelah Anda membuat titik akhir gateway, Anda dapat menambahkannya sebagai target di tabel rute Anda untuk lalu lintas yang ditujukan dari VPC Anda ke S3 Express One Zone. Seperti Amazon S3, Tidak dikenakan biaya tambahan untuk menggunakan titik akhir gateway. Untuk informasi selengkapnya tentang cara mengonfigurasi titik akhir VPC gateway, lihat Jaringan untuk S3 Express One Zone

Otorisasi berbasis sesi

Dengan S3 Express One Zone, Anda mengautentikasi dan mengotorisasi permintaan melalui mekanisme berbasis sesi baru yang dioptimalkan untuk memberikan latensi terendah. Anda dapat menggunakan `CreateSession` untuk meminta kredensial sementara yang menyediakan akses latensi rendah ke bucket Anda. Kredensial sementara ini dicakup ke bucket direktori S3 tertentu. Token sesi hanya digunakan dengan operasi Zonal (tingkat objek) (dengan pengecualian). CopyObject Untuk informasi selengkapnya, lihat `CreateSession`otorisasi.

AWS SDK yang didukung untuk S3 Express One Zone menangani pembentukan sesi dan penyegaran atas nama Anda. Untuk melindungi sesi Anda, kredensial keamanan sementara akan kedaluwarsa setelah 5 menit. Setelah Anda mengunduh dan menginstal AWS SDK dan mengonfigurasi izin yang diperlukan AWS Identity and Access Management (IAM), Anda dapat segera mulai menggunakan operasi API.

Fitur S3 Express One Zone

Fitur S3 berikut tersedia untuk S3 Express One Zone. Untuk mengetahui daftar lengkap operasi API yang didukung dan fitur yang tidak didukung, lihat. Apa yang membuat S3 Express One Zone berbeda?

Manajemen akses dan keamanan

Dengan bucket direktori, Anda dapat menggunakan fitur berikut untuk mengaudit dan mengelola akses. Secara default, bucket direktori bersifat pribadi dan hanya dapat diakses oleh pengguna yang secara eksplisit diberikan akses. Tidak seperti bucket tujuan umum, yang dapat mengatur batas kendali akses pada tingkat bucket, prefiks, atau tag objek, batas kendali akses untuk bucket direktori ditetapkan hanya pada tingkat bucket. Untuk informasi selengkapnya, lihat AWS Identity and Access Management (IAM) untuk S3 Express One Zone.

- S3 Block Public Access — Semua pengaturan Akses Publik Blok S3 diaktifkan secara default di tingkat bucket. Pengaturan default ini tidak dapat diubah.
- Kepemilikan Objek S3 (pemilik bucket diberlakukan secara default) — Daftar kontrol akses (ACL) tidak didukung untuk bucket direktori. Bucket direktori secara otomatis menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek S3. Pemilik bucket diberlakukan berarti ACL dinonaktifkan, dan pemilik bucket secara otomatis memiliki dan memiliki kontrol penuh atas setiap objek di bucket. Pengaturan default ini tidak dapat diubah.
- AWS Identity and Access Management (IAM) — IAM membantu Anda mengontrol akses ke bucket direktori Anda dengan aman. Anda dapat menggunakan IAM untuk memberikan akses ke operasi API manajemen bucket (Regional) dan operasi API manajemen objek (Zonal) melalui tindakan. `s3express:CreateSession` Untuk informasi selengkapnya, lihat AWS Identity and Access Management (IAM) untuk S3 Express One Zone. Tidak seperti tindakan manajemen objek, tindakan manajemen bucket tidak dapat dilakukan lintas akun. Hanya pemilik bucket yang dapat melakukan tindakan tersebut.
- Kebijakan bucket–Gunakan bahasa kebijakan berbasis IAM untuk mengonfigurasi izin berbasis sumber daya untuk bucket direktori Anda. Anda juga dapat menggunakan IAM untuk mengontrol akses ke operasi `CreateSession` API, yang memungkinkan Anda menggunakan operasi API Zonal, atau manajemen objek. Anda dapat memberikan akses akun yang sama atau lintas akun ke operasi API Zonal. Untuk informasi selengkapnya tentang izin dan kebijakan S3 Express One Zone, lihat. AWS Identity and Access Management (IAM) untuk S3 Express One Zone
- IAM Access Analyzer for S3 — Evaluasi dan pantau kebijakan akses Anda untuk memastikan bahwa kebijakan tersebut hanya menyediakan akses yang dimaksudkan ke sumber daya S3 Anda.

Pencatatan dan pemantauan

S3 Express One Zone menggunakan alat pencatatan dan pemantauan S3 berikut yang dapat Anda gunakan untuk memantau dan mengontrol bagaimana sumber daya Anda digunakan:

- CloudWatch Metrik Amazon — Pantau AWS sumber daya dan aplikasi Anda dengan menggunakan CloudWatch untuk mengumpulkan dan melacak metrik. S3 Express One Zone menggunakan CloudWatch namespace yang sama dengan kelas penyimpanan Amazon S3 lainnya (`AWS/S3`) dan mendukung metrik penyimpanan harian untuk bucket direktori: dan. `BucketSizeBytes` `NumberOfObjects` Untuk informasi selengkapnya, lihat Memantau metrik dengan Amazon CloudWatch.
- AWS CloudTrail log — AWS CloudTrail adalah sebuah log Layanan AWS yang membantu Anda menerapkan audit operasional dan risiko, tata kelola, dan kepatuhan Anda Akun AWS dengan mencatat tindakan yang diambil oleh pengguna, peran, atau. Layanan AWS Untuk S3 Express One Zone, CloudTrail menangkap operasi API endpoint Regional (misalnya, `CreateBucket` dan`PutBucketPolicy`) sebagai peristiwa manajemen. Peristiwa ini mencakup tindakan yang diambil dalam operasi AWS Management Console, AWS Command Line Interface (AWS CLI), AWS SDK, dan AWS API. Acara `eventsource` untuk CloudTrail manajemen untuk S3 Express One Zone adalah`s3express.amazonaws.com`. Untuk informasi selengkapnya, lihat Acara Amazon S3 CloudTrail .

> **Note**
>
> Log akses server Amazon S3 tidak didukung dengan S3 Express One Zone.

Manajemen objek

Setelah membuat bucket direktori, Anda dapat mengelola penyimpanan objek menggunakan konsol Amazon S3, AWS SDK, dan. AWS CLI Fitur-fitur berikut tersedia untuk manajemen objek dengan S3 Express One Zone:

- Operasi Batch S3 — Gunakan Operasi Batch untuk melakukan operasi massal pada objek dalam bucket direktori, misalnya, fungsi Copy dan AWS Lambda Invoke. Misalnya, Anda dapat menggunakan Operasi Batch untuk menyalin objek antara bucket direktori dan bucket tujuan umum. Dengan Operasi Batch, Anda dapat mengelola miliaran objek dalam skala besar dengan satu permintaan S3 dengan menggunakan AWS SDK atau AWS CLI atau beberapa klik di konsol Amazon S3.
- Impor–Setelah membuat bucket direktori, Anda dapat mengisi bucket dengan objek menggunakan fitur impor di konsol Amazon S3. Impor adalah metode yang disederhanakan untuk membuat pekerjaan Operasi Batch guna menyalin objek dari bucket tujuan umum ke bucket direktori.

AWS SDK dan pustaka klien

Setelah membuat bucket direktori dan mengunggah objek ke bucket, Anda dapat mengelola penyimpanan objek dengan menggunakan yang berikut ini.

- Mountpoint untuk Amazon S3 - Mountpoint untuk Amazon S3 adalah klien file sumber terbuka yang memberikan akses throughput tinggi, menurunkan biaya komputasi untuk data lake di Amazon S3. Mountpoint untuk Amazon S3 menerjemahkan panggilan API sistem file lokal ke panggilan API objek S3 seperti dan. `GET` `LIST` Ini sangat ideal untuk beban kerja data lake read-heavy yang memproses petabyte data dan membutuhkan throughput elastis tinggi yang disediakan oleh Amazon S3 untuk meningkatkan dan menurunkan di ribuan instance.
- S3A- S3A adalah antarmuka yang Hadoop kompatibel yang direkomendasikan untuk mengakses penyimpanan data di Amazon S3. S3Amenggantikan klien sistem S3N Hadoop file.
- PyTorchon AWS — PyTorch on AWS adalah kerangka pembelajaran mendalam sumber terbuka yang membuatnya lebih mudah untuk mengembangkan model pembelajaran mesin dan menerapkannya ke produksi.
- AWS SDK — Anda dapat menggunakan AWS SDK saat mengembangkan aplikasi dengan Amazon S3. AWS SDK menyederhanakan tugas pemrograman Anda dengan membungkus API Amazon S3 REST yang mendasarinya. Untuk informasi selengkapnya tentang penggunaan AWS SDK dengan S3 Express One Zone, lihat. the section called “AWS SDK”

Enkripsi dan perlindungan data

Objek yang disimpan dalam bucket direktori secara otomatis dienkripsi dengan menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3). Bucket direktori tidak mendukung enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS), enkripsi sisi server dengan kunci enkripsi yang disediakan pelanggan (SSE-C), atau enkripsi sisi server dua lapis dengan (DSSE-KMS). AWS KMS keys Untuk informasi selengkapnya, lihat Perlindungan data dan enkripsi dan Menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3).

S3 Express One Zone menawarkan kepada Anda opsi untuk memilih algoritma checksum yang digunakan untuk memvalidasi data Anda selama mengunggah atau mengunduh. Anda dapat memilih salah satu algoritma pemeriksaan integritas data Secure Hash Algorithms (SHA) atau Cyclic Redundancy Check (CRC) berikut: CRC32, CRC32C, SHA-1, dan SHA-256. Checksum berbasis MD5 tidak didukung dengan kelas penyimpanan S3 Express One Zone.

Untuk informasi selengkapnya, lihat Praktik terbaik checksum tambahan S3.

AWS Versi Tanda Tangan 4 (SigV4)

S3 Express One Zone menggunakan AWS Signature Versi 4 (SigV4). SigV4adalah protokol penandatanganan yang digunakan untuk mengautentikasi permintaan ke Amazon S3 melalui HTTPS. S3 Express One Zone menandatangani permintaan dengan menggunakan AWS Sigv4. Untuk informasi selengkapnya, lihat Mengautentikasi Permintaan (Versi AWS Tanda Tangan 4) di Referensi API Amazon Simple Storage Service.

Konsistensi kuat

S3 Express One Zone memberikan read-after-write konsistensi yang kuat untuk `PUT` dan `DELETE` permintaan objek di bucket direktori Anda secara keseluruhan. Wilayah AWS Untuk informasi selengkapnya, lihat Model konsistensi data Amazon S3.

Layanan terkait

Anda dapat menggunakan yang berikut ini Layanan AWS dengan kelas penyimpanan S3 Express One Zone untuk mendukung kasus penggunaan latensi rendah spesifik Anda.

- Amazon Elastic Compute Cloud (Amazon EC2) — Amazon EC2 menyediakan kapasitas komputasi yang aman dan dapat diskalakan di. AWS Cloud Menggunakan Amazon EC2 mengurangi kebutuhan Anda untuk berinvestasi pada perangkat keras di awal, sehingga Anda dapat mengembangkan dan mendeploy aplikasi lebih cepat. Anda dapat menggunakan Amazon EC2 untuk meluncurkan server virtual sebanyak atau sesedikit yang Anda butuhkan, mengonfigurasi keamanan dan jaringan, serta mengelola penyimpanan.
- AWS Lambda–Lambda adalah layanan komputasi yang memungkinkan Anda menjalankan kode tanpa perlu menyediakan atau mengelola server. Anda mengonfigurasi pengaturan pemberitahuan di bucket, dan memberikan izin kepada Amazon S3 untuk memanggil fungsi di kebijakan izin berbasis sumber daya milik fungsi.
- Amazon Elastic Kubernetes Service (Amazon EKS) — Amazon EKS adalah layanan terkelola yang menghilangkan kebutuhan untuk menginstal, mengoperasikan, dan memelihara Kubernetes pesawat kontrol Anda sendiri. AWSKubernetesadalah sistem sumber terbuka yang mengotomatiskan manajemen, penskalaan, dan penyebaran aplikasi kontainer.
- Amazon Elastic Container Service (Amazon ECS)—Amazon ECS adalah layanan orkestrasi kontainer yang dikelola sepenuhnya yang membantu Anda mendeploy, mengelola, dan menskalakan aplikasi kontainer dengan mudah.

- Amazon Athena–Athena adalah layanan kueri interaktif yang memudahkan analisis data di Amazon S3 dengan menggunakan SQL standar. Anda juga dapat menggunakan Athena untuk menjalankan analisis data secara interaktif dengan menggunakan Apache Spark tanpa harus merencanakan, mengonfigurasi, atau mengelola sumber daya. Ketika Anda menjalankan Apache Spark aplikasi di Athena, Anda mengirimkan Spark kode untuk diproses dan menerima hasilnya secara langsung.
- Pelatihan Model SageMaker Runtime Amazon — Amazon SageMaker Runtime adalah layanan pembelajaran mesin yang dikelola sepenuhnya. Dengan SageMaker Runtime, ilmuwan dan pengembang data dapat dengan cepat dan mudah membangun dan melatih model pembelajaran mesin, dan kemudian langsung menerapkannya ke lingkungan host yang siap produksi.
- AWS Glue— AWS Glue adalah layanan integrasi data tanpa server yang memudahkan pengguna analitik untuk menemukan, menyiapkan, memindahkan, dan mengintegrasikan data dari berbagai sumber. Anda dapat menggunakannya AWS Glue untuk analitik, pembelajaran mesin, dan pengembangan aplikasi. AWS Glue juga mencakup produktivitas tambahan dan perkakas operasi data untuk menulis, menjalankan pekerjaan, dan mengimplementasikan alur kerja bisnis.
- Amazon EMR - Amazon EMR adalah platform cluster terkelola yang menyederhanakan menjalankan kerangka kerja data besar, seperti Apache Hadoop danApache Spark, AWS untuk memproses dan menganalisis sejumlah besar data.

Langkah selanjutnya

Untuk informasi selengkapnya tentang bekerja dengan kelas penyimpanan dan bucket direktori S3 Express One Zone, lihat topik berikut:

- Apa yang membuat S3 Express One Zone berbeda?
- Memulai dengan S3 Express One Zone
- Jaringan untuk S3 Express One Zone
- Ember direktori
- Bekerja dengan objek dalam ember direktori
- Keamanan untuk S3 Express One Zone
- Mengoptimalkan Kinerja Amazon S3 Express One Zone
- Mengembangkan dengan S3 Express One Zone

Apa yang membuat S3 Express One Zone berbeda?

Amazon S3 Express One Zone adalah kelas penyimpanan Amazon S3 tunggal berkinerja tinggi yang dibuat khusus untuk menghadirkan akses data milidetik satu digit yang konsisten untuk aplikasi Anda yang paling sensitif terhadap latensi. S3 Express One Zone adalah kelas penyimpanan S3 pertama di mana Anda dapat memilih satu Zona Ketersediaan dengan opsi untuk menempatkan bersama penyimpanan objek dengan sumber daya komputasi Anda, yang memberikan kecepatan akses setinggi mungkin. Selain itu, untuk lebih meningkatkan kecepatan akses dan mendukung ratusan ribu permintaan per detik, data S3 Express One Zone disimpan dalam jenis bucket baru: bucket direktori Amazon S3.

Untuk informasi selengkapnya, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Anda dapat membuat bucket direktori dan mengakses data Anda di S3 Express One Zone dengan menggunakan API Amazon S3. API Amazon S3 kompatibel dengan S3 Express One Zone dan bucket direktori, kecuali beberapa perbedaan yang mencolok. Untuk informasi tentang perbedaan S3 Express One Zone, lihat topik berikut.

Topik

- Perbedaan S3 Express One Zone
- Operasi API yang didukung oleh S3 Express One Zone
- Fitur Amazon S3 tidak didukung oleh S3 Express One Zone

Perbedaan S3 Express One Zone

- Jenis bucket yang didukung—Objek di kelas penyimpanan S3 Express One Zone hanya dapat disimpan dalam bucket direktori. Untuk informasi selengkapnya, lihat Ember direktori.
- Daya Tahan—Dengan S3 Express One Zone, data Anda disimpan secara berlebihan di beberapa perangkat dalam satu Zona Ketersediaan. S3 Express One Zone dirancang untuk ketersediaan sebesar 99,95% dalam satu Zona Ketersediaan dan didukung oleh Perjanjian Tingkat Layanan Amazon S3. Untuk informasi selengkapnya, lihat Zona Ketersediaan Tunggal.
- **`ListObjectsV2`**perilaku
  - Untuk bucket direktori, `ListObjectsV2` tidak mengembalikan objek dalam urutan leksikografis (abjad). Selain itu, prefiks harus diakhiri dengan pembatas dan hanya “/” yang dapat ditentukan sebagai pembatas.

  - Untuk bucket direktori, `ListObjectsV2` respons menyertakan awalan yang hanya terkait dengan unggahan multibagian yang sedang berlangsung.
- Perilaku penghapusan—Saat Anda menghapus objek di bucket direktori, Amazon S3 secara rekursif menghapus direktori kosong apa pun di jalur objek. Misalnya, jika Anda menghapus kunci objek`dir1/dir2/file1.txt`, Amazon S3 menghapus. `file1.txt` Jika direktori `dir1/` dan `dir2/` kosong dan tidak berisi objek lain, Amazon S3 juga menghapus direktori tersebut.
- ETag dan checksum—Tanda entitas (ETag) untuk S3 Express One Zone adalah string alfanumerik acak dan bukan checksum MD5. Untuk informasi selengkapnya tentang penggunaan checksum tambahan dengan S3 Express One Zone, lihat Praktik terbaik checksum tambahan S3.
- Kunci objek dalam permintaan **`DeleteObjects`**
  - Kunci objek dalam permintaan `DeleteObjects` harus berisi setidaknya satu karakter spasi non-putih. String dari semua karakter spasi putih tidak didukung dalam permintaan `DeleteObjects`.
  - Kunci objek dalam permintaan `DeleteObjects` tidak dapat berisi karakter kontrol Unicode, kecuali karakter baris baru (`\n`), tab (`\t`), dan kembali ke awal (`\r`).
- Titik akhir Regional dan Zonal—Saat menggunakan S3 Express One Zone, Anda harus menentukan Wilayah di semua permintaan klien. Untuk titik akhir Regional, Anda perlumenentukan Wilayah, misalnya `s3express-control.us-west-2.amazonaws.com`. Untuk titik akhir Zonal, Anda menentukan Wilayah dan Zona Ketersediaan, misalnya `s3express-usw2-az1.us-west-2.amazonaws.com`. Untuk informasi selengkapnya, lihat Titik akhir Regional dan Zona.
- Unggahan multi-bagian–Seperti objek lain yang disimpan di Amazon S3, Anda dapat mengunggah dan menyalin objek besar yang disimpan di kelas penyimpanan S3 Express One Zone dengan menggunakan proses unggahan multi-bagian. Namun, berikut ini adalah beberapa perbedaan saat menggunakan proses pengunggahan multi-bagian dengan objek yang disimpan di S3 Express One Zone. Untuk informasi selengkapnya, lihat the section called “Menggunakan unggahan multibagian dengan bucket direktori”.
  - Tanggal pembuatan objek adalah tanggal penyelesaian unggahan multi-bagian.
  - Nomor bagian multi-bagian harus menggunakan nomor bagian berurutan. Jika Anda mencoba menyelesaikan permintaan unggahan multi-bagian dengan nomor bagian yang tidak berurutan, Amazon S3 akan menampilkan kesalahan HTTP `400 (Bad Request)`.
  - Pemrakarsa unggahan multi-bagian dapat membatalkan permintaan unggahan multi-bagian hanya jika mereka telah diberi izin eksplisit `AbortMultipartUpload` untuk mengakses melalui izin `s3express:CreateSession`. Untuk informasi selengkapnya, lihat AWS Identity and Access Management (IAM) untuk S3 Express One Zone.

- Mengosongkan bucket direktori — `s3 rm` Perintah melalui (AWS Command Line Interface CLI), `delete` operasi melalui Mountpoint, dan tombol opsi Empty bucket melalui tombol tidak dapat menghapus unggahan multipart AWS Management Console yang sedang berlangsung dalam bucket direktori. Untuk menghapus unggahan multibagian yang sedang berlangsung ini, gunakan `ListMultipartUploads` operasi untuk mencantumkan unggahan multibagian yang sedang berlangsung di bucket dan gunakan `AbortMultupartUpload` operasi untuk membatalkan semua unggahan multibagian yang sedang berlangsung.

Operasi API yang didukung oleh S3 Express One Zone

Kelas penyimpanan Amazon S3 Express One Zone mendukung operasi API titik akhir Regional (tingkat bucket, atau bidang kontrol) dan Zonal (level objek, atau bidang data). Untuk informasi selengkapnya, lihat Jaringan untuk S3 Express One Zone dan Titik akhir dan titik akhir VPC gateway.

Operasi API titik akhir regional

Operasi API titik akhir Regional berikut didukung untuk S3 Express One Zone:

- CreateBucket
- DeleteBucket
- DeleteBucketPolicy
- GetBucketPolicy
- ListDirectoryBuckets
- PutBucketPolicy

Operasi API titik akhir zonal

Operasi API titik akhir Zonal berikut didukung untuk S3 Express One Zone:

- CreateSession
- CopyObject
- DeleteObject
- DeleteObjects
- GetObject
- GetObjectAttributes
- HeadBucket

- HeadObject
- ListObjectsV2
- PutObject
- AbortMultipartUpload
- CompleteMultiPartUpload
- CreateMultipartUpload
- ListMultipartUploads
- ListParts
- UploadPart
- UploadPartCopy

Fitur Amazon S3 tidak didukung oleh S3 Express One Zone

Fitur Amazon S3 berikut tidak didukung oleh S3 Express One Zone:

- AWS CloudTrail peristiwa pesawat data
- AWS kebijakan terkelola
- AWS PrivateLink untuk S3
- Checksum MD5
- Hapus autentikasi multi-faktor (MFA)
- Kunci Objek S3
- Pembayaran oleh Pemohon
- Pemberian Akses S3
- Titik Akses S3
- Tanda bucket
- Metrik CloudWatch permintaan Amazon
- Notifikasi Peristiwa S3
- Siklus Hidup S3
- Titik Akses Multi-Wilayah S3
- Titik Akses Lambda Objek S3
- Penentuan Versi S3
- Inventaris S3

- Replikasi S3
- Tanda objek
- Pilih S3
- Log akses server
- Hosting situs web statis
- Lensa Penyimpanan S3
- Grup Lensa Penyimpanan S3
- Akselerasi Transfer S3
- Enkripsi sisi server dua lapis dengan kunci () (DSSE-KMS AWS Key Management Service)AWS KMS
- Enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS)
- Enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C)
- Opsi untuk menyalin pengaturan bucket yang ada saat membuat bucket baru AWS Management Console.

Memulai dengan S3 Express One Zone

Bagian berikut menjelaskan cara memulai menggunakan kelas penyimpanan Amazon S3 Express One Zone dan bucket direktori. Untuk informasi selengkapnya, lihat Apa itu S3 Express One Zone?.

Topik

- Siapkan AWS Identity and Access Management (IAM) dengan S3 Express One Zone
- Konfigurasikan titik akhir VPC gateway
- Bekerja dengan S3 Express One Zone dengan menggunakan konsol S3, AWS CLI, dan SDK AWS

Siapkan AWS Identity and Access Management (IAM) dengan S3 Express One Zone

AWS Identity and Access Management (IAM) adalah sebuah Layanan AWS yang membantu administrator mengontrol akses ke sumber daya dengan aman. AWS Administrator IAM mengendalikan siapa saja yang dapat diautentikasi (masuk) dan diizinkan (memiliki izin) untuk menggunakan sumber daya Amazon S3 di S3 Express One Zone. Anda dapat menggunakan IAM tanpa biaya tambahan.

Secara default, pengguna tidak memiliki izin untuk direktori bucket dan operasi S3 Express One Zone. Untuk memberikan izin akses ke direktori bucket dan operasi S3 Express One Zone, Anda bisa menggunakan IAM untuk membuat pengguna atau peran dan melampirkan izin ke identitas tersebut.

Untuk memulai dengan IAM, lihat AWS Identity and Access Management (IAM) untuk S3 Express One Zone danKebijakan berbasis identitas IAM untuk S3 Express One Zone.

Konfigurasikan titik akhir VPC gateway

Untuk mengakses S3 Express One Zone, Anda menggunakan titik akhir Regional dan Zonal yang berbeda dari titik akhir Amazon S3 standar. Bergantung pada operasi Amazon S3 API yang Anda gunakan, titik akhir Zonal atau Regional diperlukan. Untuk daftar lengkap operasi API yang didukung menurut jenis titik akhir, lihat Operasi API yang didukung oleh S3 Express One Zone. Anda harus mengakses titik akhir Zonal dan Regional melalui titik akhir gateway cloud privat virtual (VPC). Untuk mengonfigurasi titik akhir gateway, lihatJaringan untuk S3 Express One Zone.

Bekerja dengan S3 Express One Zone dengan menggunakan konsol S3, AWS CLI, dan SDK AWS

Anda dapat bekerja dengan kelas penyimpanan S3 Express One Zone dan bucket direktori dengan menggunakan AWS SDK, konsol Amazon S3, AWS Command Line Interface (),AWS CLI dan Amazon S3 REST API.

Konsol S3

Untuk mulai menggunakan konsol S3, ikuti langkah-langkah ini:

- Membuat bucket direktori
- Mengosongkan bucket direktori
- Menghapus bucket direktori

AWS SDK

S3 Express One Zone mendukung AWS SDK berikut:

- AWS SDK for C++
- AWS SDK for Go v2

- AWS SDK for Java 2.x
- AWS SDK for JavaScript v3
- AWS SDK for .NET
- AWS SDK for PHP
- AWS SDK for Python (Boto3)
- AWS SDK for Ruby
- AWS SDK for Kotlin
- AWS SDK for Rust

Saat Anda bekerja dengan S3 Express One Zone, sebaiknya gunakan SDK AWS versi terbaru. AWS SDK yang didukung untuk S3 Express One Zone menangani pembentukan sesi, penyegaran, dan penghentian atas nama Anda. Ini berarti Anda dapat segera mulai menggunakan operasi API setelah mengunduh dan menginstal AWS SDK dan mengonfigurasi izin IAM yang diperlukan. Untuk informasi selengkapnya, lihat AWS Identity and Access Management (IAM) untuk S3 Express One Zone.

Untuk informasi tentang AWS SDK, termasuk cara mengunduh dan menginstalnya, lihat Alat untuk Dibangun AWS.

Untuk contoh AWS SDK, lihat berikut ini:

- Membuat bucket direktori
- Mengosongkan bucket direktori
- Menghapus bucket direktori

AWS Command Line Interface (AWS CLI)

Anda dapat menggunakan AWS Command Line Interface (AWS CLI) untuk membuat bucket direktori dan menggunakan operasi API titik akhir Regional dan Zonal yang didukung untuk S3 Express One Zone.

Untuk memulai AWS CLI, lihat Memulai dengan AWS CLI di AWS CLI Command Reference.

> **Note**
>
> Untuk menggunakan bucket direktori dengan `aws s3`perintah tingkat tinggi, perbarui AWS CLI ke versi terbaru. Untuk informasi selengkapnya tentang cara menginstal dan

mengonfigurasi AWS CLI, lihat Menginstal atau memperbarui versi terbaru dari Referensi AWS CLI Perintah. AWS CLI

Sebagai AWS CLI contoh, lihat yang berikut ini:

- Membuat bucket direktori
- Mengosongkan bucket direktori
- Menghapus bucket direktori

Jaringan untuk S3 Express One Zone

Untuk mengakses objek kelas penyimpanan Amazon S3 Express One Zone dan bucket direktori, Anda menggunakan titik akhir API Regional dan Zonal yang berbeda dari titik akhir Amazon S3 standar. Bergantung pada operasi API S3 yang Anda gunakan, titik akhir Zonal atau Regional diperlukan. Untuk daftar lengkap operasi API menurut jenis titik akhir, lihat Operasi API yang didukung oleh S3 Express One Zone.

Anda dapat mengakses operasi API Zonal dan Regional melalui titik akhir cloud privat virtual (VPC) gateway. Untuk mengonfigurasi titik akhir VPC gateway, lihat the section called “Mengkonfigurasi titik akhir gateway VPC”.

Topik berikut menjelaskan persyaratan jaringan untuk mengakses S3 Express One Zone dengan menggunakan titik akhir VPC gateway.

Topik

- Titik akhir
- Mengkonfigurasi titik akhir gateway VPC

Titik akhir

Anda dapat mengakses objek kelas penyimpanan Amazon S3 Express One Zone dan bucket direktori dari VPC Anda dengan menggunakan titik akhir VPC gateway. S3 Express One Zone menggunakan titik akhir API Regional dan Zonal. Bergantung pada operasi Amazon S3 API yang Anda gunakan, titik akhir Regional atau Zonal diperlukan. Tidak dikenakan biaya tambahan untuk menggunakan titik akhir gateway.

Operasi API tingkat bucket (atau bidang kontrol) tersedia melalui titik akhir Regional dan disebut sebagai operasi API titik akhir Regional. Contoh operasi API titik akhir Regional adalah `CreateBucket` dan `DeleteBucket`. Ketika membuat direktori bucket, pilihlah satu Ketersediaan tempat direktori bucket akan dibuat. Setelah membuat bucket direktori, Anda dapat menggunakan operasi API titik akhir Zonal untuk mengunggah dan mengelola objek di bucket direktori.

Operasi API tingkat objek (atau bidang data) tersedia melalui titik akhir Zonal dan disebut sebagai operasi API titik akhir Zonal. Contoh operasi API titik akhir Zonal adalah `CreateSession` dan `PutObject`.

Tabel berikut menunjukkan titik akhir API Regional dan Zonal yang tersedia untuk setiap Wilayah dan Zona Ketersediaan.

Mengkonfigurasi titik akhir gateway VPC

Gunakan prosedur berikut untuk membuat titik akhir gateway yang terhubung ke objek kelas penyimpanan Amazon S3 Express One Zone dan bucket direktori.

Untuk mengonfigurasi titik akhir VPC gateway

1. Buka Konsol VPC Amazon di https://console.aws.amazon.com/vpc/.
2. Di panel navigasi, pilih Titik akhir.
3. Pilih Buat Titik Akhir.
4. Buat nama untuk titik akhir Anda.
5. Untuk Kategori layanan, pilih Layanan AWS.
6. Untuk Layanan, tambahkan filter Type=Gateway dan kemudian pilih tombol opsi di sebelah com.amazonaws.***region*** .s3express.
7. Untuk VPC, pilih VPC yang akan digunakan untuk menciptakan titik akhir.
8. Untuk Tabel rute, pilih tabel rute yang akan digunakan oleh titik akhir. Amazon VPC secara otomatis menambahkan rute yang mengarahkan lalu lintas yang ditujukan untuk layanan ke antarmuka jaringan titik akhir.
9. Untuk Kebijakan, pilih Akses penuh untuk mengizinkan semua operasi oleh semua pengguna utama pada semua sumber daya melalui titik akhir VPC. Jika tidak, pilih Kustom untuk melampirkan kebijakan titik akhir VPC yang mengontrol izin yang dimiliki pengguna utama untuk melakukan tindakan pada sumber daya melalui titik akhir VPC.
10. (Opsional) Untuk menambahkan tanda, pilih Tambahkan tanda baru dan masukkan kunci dan nilai tanda.

11. Pilih Buat Titik Akhir.

Setelah membuat titik akhir gateway, Anda dapat menggunakan titik akhir API Regional dan titik akhir API Zonal untuk mengakses objek kelas penyimpanan Amazon S3 Express One Zone dan bucket direktori.

Ember direktori

Terdapat dua tipe bucket Amazon S3, bucket bertujuan umum dan bucket direktori. Pilih tipe bucket yang paling sesuai dengan aplikasi dan persyaratan kinerja Anda:

- Bucket tujuan umum adalah jenis bucket S3 asli dan direkomendasikan untuk sebagian besar kasus penggunaan dan pola akses. Bucket tujuan umum juga memungkinkan objek yang disimpan di semua kelas penyimpanan, kecuali S3 Express One Zone.
- Bucket direktori menggunakan kelas penyimpanan S3 Express One Zone, yang direkomendasikan jika aplikasi Anda sensitif terhadap kinerja dan mendapat manfaat dari latensi `PUT` dan `GET` satu digit milidetik.

Bucket direktori digunakan untuk beban kerja atau aplikasi kritis kinerja yang memerlukan latensi satu digit milidetik yang konsisten. Bucket direktori mengatur data secara hierarkis ke dalam direktori, alih-alih struktur penyimpanan datar dari bucket tujuan umum. Tidak ada batas prefiks untuk bucket direktori dan direktori individu dapat menskalakan secara horizontal.

Bucket direktori menggunakan kelas penyimpanan S3 Express One Zone, yang menyimpan data di beberapa perangkat dalam satu Zona Ketersediaan tetapi tidak menyimpan data secara berlebihan di seluruh Zona Ketersediaan. Saat membuat bucket direktori, sebaiknya tentukan Wilayah AWS dan Availability Zone yang lokal untuk Amazon EC2, Amazon Elastic Kubernetes Service, atau Amazon Elastic Container Service (Amazon ECS) menghitung instans untuk mengoptimalkan kinerja.

Anda dapat membuat hingga 10 ember direktori di masing-masing Akun AWS, tanpa batasan jumlah objek yang dapat Anda simpan dalam ember. Kuota bucket Anda diterapkan ke setiap Wilayah di wilayah Anda Akun AWS. Jika aplikasi Anda memerlukan peningkatan batas ini, hubungi AWS Support. Untuk informasi selengkapnya, kunjungi konsol Service Quotas.

> **Important**
>
> Bucket direktori yang tidak memiliki aktivitas permintaan untuk periode minimal 90 hari transisi ke status tidak aktif. Saat dalam keadaan tidak aktif, bucket direktori sementara

tidak dapat diakses untuk dibaca dan ditulis. Bucket yang tidak aktif menyimpan semua penyimpanan, metadata objek, dan metadata bucket. Biaya penyimpanan yang ada berlaku untuk ember yang tidak aktif. Jika Anda membuat permintaan akses ke bucket yang tidak aktif, bucket akan beralih ke status aktif, biasanya dalam beberapa menit. Selama periode transisi ini, membaca dan menulis mengembalikan kode `503 (Service Unavailable)` kesalahan HTTP.

Topik berikut menyediakan informasi tentang bucket direktori. Untuk informasi selengkapnya tentang bucket tujuan umum, lihat Gambaran umum bucket.

Topik

- Zona Ketersediaan
- Nama bucket direktori
- Direktori
- Nama kunci
- Manajemen akses
- Bekerja dengan bucket direktori
- Aturan penamaan bucket direktori
- Membuat bucket direktori
- Melihat properti bucket direktori
- Mengelola kebijakan bucket untuk bucket direktori
- Mengosongkan bucket direktori
- Menghapus bucket direktori
- Mendaftar bucket direktori
- Menggunakan HeadBucket dengan ember direktori

Zona Ketersediaan

Saat membuat bucket direktori, Anda memilih Zona Ketersediaan dan Wilayah AWS.

Bucket direktori menggunakan kelas penyimpanan S3 Express One Zone, yang dibuat untuk digunakan oleh aplikasi yang sensitif terhadap kinerja. S3 Express One Zone adalah kelas penyimpanan S3 pertama di mana Anda dapat memilih satu Zona Ketersediaan dengan opsi

untuk menempatkan bersama penyimpanan objek dengan sumber daya komputasi Anda, yang memberikan kecepatan akses setinggi mungkin.

Dengan S3 Express One Zone, data Anda disimpan secara berlebihan di beberapa perangkat dalam satu Zona Ketersediaan. S3 Express One Zone dirancang untuk ketersediaan 99,95 persen dalam satu Availability Zone dan didukung oleh Perjanjian Tingkat Layanan Amazon S3. Untuk informasi selengkapnya, lihat Zona Ketersediaan Tunggal

Nama bucket direktori

Nama bucket direktori terdiri dari nama dasar yang Anda berikan dan akhiran yang berisi ID Zona Ketersediaan tempat bucket Anda berada. Nama bucket direktori harus mengikuti format ini dan mematuhi aturan penamaan bucket direktori:

```
bucket-base-name--azid--x-s3
```

Misalnya, nama bucket direktori berikut berisi ID Zona Ketersediaan `usw2-az1`:

```
bucket-base-name--usw2-az1--x-s3
```

Untuk informasi selengkapnya, lihat Aturan penamaan bucket direktori.

Direktori

Bucket direktori mengatur data secara hierarkis ke dalam direktori, alih-alih struktur penyortiran datar dari bucket tujuan umum. Setiap bucket direktori S3 dapat mendukung ratusan ribu transaksi per detik (TPS), terlepas dari jumlah direktori dalam bucket.

Dengan namespace hierarkis, pembatas dalam kunci objek menjadi penting. Satu-satunya pembatas yang didukung adalah garis miring (`/`). Direktori ditentukan oleh batas pembatas. Misalnya, kunci objek `dir1/dir2/file1.txt` menghasilkan direktori `dir1/` dan `dir2/` secara otomatis dibuat, dan objek `file1.txt` ditambahkan ke direktori `/dir2` di jalur `dir1/dir2/file1.txt`.

Model pengindeksan bucket direktori mengembalikan hasil yang tidak disortir untuk operasi API `ListObjectsV2`. Jika Anda perlu membatasi hasil Anda ke subbagian bucket Anda, Anda dapat menentukan jalur subdirektori dalam parameter `prefix`, misalnya, `prefix=dir1/`.

Nama kunci

Untuk bucket direktori, subdirektori yang umum untuk beberapa kunci objek dibuat dengan kunci objek pertama. Kunci objek tambahan untuk subdirektori yang sama menggunakan subdirektori yang dibuat sebelumnya. Model ini memberi Anda fleksibilitas dalam memilih kunci objek yang paling cocok untuk aplikasi, dengan dukungan yang sama untuk direktori yang jarang dan padat.

Manajemen akses

Bucket direktori memiliki semua pengaturan Blokir Akses Publik S3 yang diaktifkan secara default di tingkat bucket. Kepemilikan Objek S3 diatur ke pemilik bucket yang diberlakukan dan daftar kontrol akses (ACL) dinonaktifkan. Pengaturan ini tidak dapat dimodifikasi.

Secara default, pengguna tidak memiliki izin untuk bucket direktori dan operasi S3 Express One Zone. Untuk memberikan izin akses bagi bucket direktori, Anda dapat menggunakan IAM untuk membuat pengguna, grup, atau peran dan melampirkan izin ke identitas tersebut. Untuk informasi selengkapnya, lihat AWS Identity and Access Management (IAM) untuk S3 Express One Zone.

Bekerja dengan bucket direktori

Untuk informasi selengkapnya tentang bekerja dengan bucket direktori, lihat topik berikut.

Topik

- Aturan penamaan bucket direktori
- Membuat bucket direktori
- Melihat properti bucket direktori
- Mengelola kebijakan bucket untuk bucket direktori
- Mengosongkan bucket direktori
- Menghapus bucket direktori
- Mendaftar bucket direktori
- Menggunakan HeadBucket dengan ember direktori

Aturan penamaan bucket direktori

Saat membuat bucket direktori di Amazon S3, aturan penamaan bucket berikut berlaku. Untuk aturan penamaan bucket bertujuan umum, lihat Peraturan penamaan bucket.

Nama bucket direktori terdiri dari nama dasar yang Anda berikan, dan akhiran yang berisi ID AWS Availability Zone tempat bucket Anda berada dan`--x-s3`.

```
base-name--azid--x-s3
```

Misalnya, nama bucket direktori berikut berisi ID Zona Ketersediaan `usw2-az1`:

```
bucket-base-name--usw2-az1--x-s3
```

> **Note**
>
> Saat Anda membuat bucket direktori menggunakan konsol, akhiran secara otomatis ditambahkan ke nama dasar yang Anda berikan. Sufiks ini mencakup ID Zona Ketersediaan dari Zona Ketersediaan yang Anda pilih.
> Saat membuat bucket direktori menggunakan API, Anda harus memberikan akhiran lengkap, termasuk ID Availability Zone, dalam permintaan Anda. Untuk daftar ID Availability Zone, lihatZona Ketersediaan dan Wilayah S3 Express One Zone.

Nama bucket direktori harus:

- Jadilah unik dalam Zona yang dipilih Wilayah AWS dan Availability Zone.
- Panjang nama harus antara 3 (menit) dan 63 (maks) karakter, termasuk akhiran.
- Hanya terdiri dari huruf kecil, angka, tanda hubung (-).
- Dimulai dan diakhiri dengan huruf atau angka.
- Harus menyertakan akhiran berikut:`--azid--x-s3`.

Membuat bucket direktori

Untuk mulai menggunakan kelas penyimpanan Amazon S3 Express One Zone, Anda membuat bucket direktori. Kelas penyimpanan S3 Express One Zone hanya dapat digunakan dengan bucket direktori. Kelas penyimpanan S3 Express One Zone mendukung kasus penggunaan latensi rendah dan menyediakan pemrosesan data yang lebih cepat dalam satu Zona Ketersediaan. Jika aplikasi Anda sensitif terhadap kinerja dan mendapat manfaat dari latensi `PUT` dan `GET` satu digit milidetik, sebaiknya buat bucket direktori sehingga Anda dapat menggunakan kelas penyimpanan S3 Express One Zone.

Terdapat dua tipe bucket Amazon S3, bucket bertujuan umum dan bucket direktori. Anda harus memilih tipe bucket yang paling sesuai dengan aplikasi dan persyaratan kinerja Anda. Bucket bertujuan umum adalah tipe bucket S3 asli. Bucket tujuan umum direkomendasikan untuk sebagian besar kasus penggunaan dan pola akses dan memungkinkan objek disimpan di semua kelas penyimpanan, kecuali S3 Express One Zone. Untuk informasi selengkapnya tentang bucket tujuan umum, lihat Gambaran umum bucket.

Bucket direktori menggunakan kelas penyimpanan S3 Express One Zone, yang dirancang agar digunakan untuk beban kerja atau aplikasi yang kritis kinerja yang memerlukan latensi milidetik satu digit yang konsisten. S3 Express One Zone adalah kelas penyimpanan S3 pertama di mana Anda dapat memilih satu Zona Ketersediaan dengan opsi untuk menempatkan bersama penyimpanan objek dengan sumber daya komputasi Anda, yang memberikan kecepatan akses setinggi mungkin. Saat membuat bucket direktori, Anda dapat secara opsional menentukan Wilayah AWS dan Availability Zone yang lokal ke Amazon EC2, Amazon Elastic Kubernetes Service, atau Amazon Elastic Container Service (Amazon ECS) menghitung instans untuk mengoptimalkan kinerja.

Dengan S3 Express One Zone, data Anda disimpan secara berlebihan di beberapa perangkat dalam satu Zona Ketersediaan. S3 Express One Zone dirancang untuk ketersediaan 99,95 persen dalam satu Availability Zone dan didukung oleh Perjanjian Tingkat Layanan Amazon S3. Untuk informasi selengkapnya, lihat Zona Ketersediaan Tunggal

Bucket direktori mengatur data secara hierarkis ke dalam direktori, sebagai lawan dari struktur penyimpanan datar dari bucket tujuan umum. Tidak ada batas prefiks untuk bucket direktori dan direktori individu dapat menskalakan secara horizontal.

Untuk informasi selengkapnya tentang bucket direktori, lihat Ember direktori.

Nama bucket direktori

Nama bucket direktori harus mengikuti format ini dan mematuhi aturan penamaan bucket direktori:

```
bucket-base-name--azid--x-s3
```

Misalnya, nama bucket direktori berikut berisi ID Zona Ketersediaan `usw2-az1`:

```
bucket-base-name--usw2-az1--x-s3
```

Untuk informasi selengkapnya tentang aturan penamaan bucket direktori, lihat Aturan penamaan bucket direktori.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di bilah navigasi di bagian atas halaman, pilih nama yang saat ini ditampilkan Wilayah AWS. Selanjutnya, pilih Wilayah tempat Anda ingin membuat ember.

   > **Note**
   >
   > Untuk meminimalkan latensi dan biaya serta memenuhi persyaratan regulasi, pilih Wilayah yang dekat dengan Anda. Objek yang disimpan di Wilayah tidak pernah keluar dari Wilayah kecuali Anda secara tegas mentransfer atau mereplikasinya ke Wilayah lain. Untuk daftar Amazon S3 Wilayah AWS, lihat Layanan AWS titik akhir di. Referensi Umum Amazon Web Services

3. Di panel navigasi kiri, pilih Bucket.
4. Pilih Buat bucket.

   Halaman Buat bucket terbuka.

5. Di bawah Konfigurasi umum, lihat Wilayah AWS tempat bucket Anda akan dibuat.
6. Di bawah Jenis Bucket, pilih Direktori.

   > **Note**
   >
   > - Jika Anda memilih Region yang tidak mendukung bucket direktori, opsi jenis Bucket akan hilang, dan tipe bucket default ke bucket tujuan umum. Untuk membuat bucket direktori, Anda harus memilih Region yang didukung. Untuk daftar Wilayah yang mendukung bucket direktori dan kelas penyimpanan Amazon S3 Express One Zone, lihat. the section called “Zona Ketersediaan dan Wilayah S3 Express One Zone”
   > - Setelah membuat bucket, Anda tidak dapat mengubah tipe bucket.

   Untuk Zona Ketersediaan, pilih Zona Ketersediaan lokal untuk layanan komputasi Anda. Untuk daftar Availability Zones yang mendukung bucket direktori dan kelas penyimpanan S3 Express One Zone, lihat. the section called “Zona Ketersediaan dan Wilayah S3 Express One Zone”

> **Note**
>
> Zona Ketersediaan tidak dapat diubah setelah bucket dibuat.

7. Di bawah Zona Ketersediaan, pilih kotak centang untuk mengetahui bahwa jika terjadi pemadaman Zona Ketersediaan, data Anda mungkin tidak tersedia atau hilang.

> **Important**
>
> Meskipun bucket direktori disimpan di beberapa perangkat dalam satu Availability Zone, bucket direktori tidak menyimpan data secara berlebihan di Availability Zone.

8. Untuk Nama bucket, masukkan nama untuk bucket direktori Anda.

   Nama bucket direktori harus:

   - Jadilah unik dalam Zona yang dipilih Wilayah AWS dan Availability Zone.
   - Panjang nama harus antara 3 (menit) dan 63 (maks) karakter, termasuk akhiran.
   - Hanya terdiri dari huruf kecil, angka, tanda hubung (-).
   - Dimulai dan diakhiri dengan huruf atau angka.
   - Harus menyertakan akhiran berikut:`--azid--x-s3`.

   Sufiks secara otomatis ditambahkan ke nama dasar yang Anda berikan saat membuat bucket direktori menggunakan konsol. Sufiks ini mencakup ID Zona Ketersediaan dari Zona Ketersediaan yang Anda pilih.

   Setelah membuat bucket, Anda tidak dapat mengubah namanya. Untuk informasi selengkapnya tentang penamaan bucket, lihat Peraturan penamaan bucket.

> **Important**
>
> Jangan sertakan informasi sensitif, seperti nomor akun, dalam nama bucket. Nama bucket terlihat dalam URL yang menunjuk objek dalam bucket.

9. Di bawah Kepemilikan Objek, setelan yang diberlakukan pemilik Bucket diaktifkan secara otomatis, dan semua daftar kontrol akses (ACL) dinonaktifkan. Untuk bucket direktori, ACL tidak dapat diaktifkan.

ACL dinonaktifkan

- Pemilik bucket yang diberlakukan (default)–ACL dinonaktifkan, dan pemilik bucket secara otomatis memilikinya, serta mendapatkan kontrol penuh atas setiap objek di dalam bucket. ACL tidak lagi memengaruhi izin akses ke data di bucket S3. Bucket menggunakan kebijakan secara eksklusif untuk menentukan kontrol akses.

  Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

10. Di bawah pengaturan Blokir Akses Publik untuk bucket ini, semua setelan Blokir Akses Publik untuk bucket direktori Anda diaktifkan secara otomatis. Pengaturan ini tidak dapat dimodifikasi untuk bucket direktori. Untuk informasi lebih lanjut tentang pemblokiran akses publik, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.
11. Di bawah pengaturan enkripsi sisi server, Amazon S3 menerapkan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi untuk semua bucket S3. Semua unggahan objek ke bucket direktori dienkripsi dengan SSE-S3. Untuk bucket direktori, jenis enkripsi tidak dapat dimodifikasi. Untuk informasi selengkapnya tentang SSE-S3, lihat. the section called “Kunci enkripsi terkelola Amazon S3 (SSE-S3)”
12. Pilih Buat bucket.

    Setelah membuat bucket, Anda dapat menambahkan file dan folder ke bucket. Untuk informasi selengkapnya, lihat the section called “Bekerja dengan objek di dalam bucket direktori”.

Menggunakan AWS SDK

SDK for Go

Contoh ini menunjukkan cara membuat bucket direktori dengan menggunakan file AWS SDK for Go.

Example

```
var bucket = "..."

func runCreateBucket(c *s3.Client) {
    resp, err := c.CreateBucket(context.Background(), &s3.CreateBucketInput{
        Bucket: &bucket,
```

```
        CreateBucketConfiguration: &types.CreateBucketConfiguration{
            Location: &types.LocationInfo{
                Name: aws.String("usw2-az1"),
                Type: types.LocationTypeAvailabilityZone,
            },
            Bucket: &types.BucketInfo{
                DataRedundancy: types.DataRedundancySingleAvailabilityZone,
                Type:           types.BucketTypeDirectory,
            },
        },
    })
    var terr *types.BucketAlreadyOwnedByYou
    if errors.As(err, &terr) {
        fmt.Printf("BucketAlreadyOwnedByYou: %s\n", aws.ToString(terr.Message))
        fmt.Printf("noop...\n")
        return
    }
    if err != nil {
        log.Fatal(err)
    }

    fmt.Printf("bucket created at %s\n", aws.ToString(resp.Location))
}
```

SDK for Java 2.x

Contoh ini menunjukkan cara membuat bucket direktori dengan menggunakan file AWS SDK for Java 2.x.

Example

```
public static void createBucket(S3Client s3Client, String bucketName) {

    //Bucket name format is {base-bucket-name}--{az-id}--x-s3
    //example: doc-example-bucket--usw2-az1--x-s3 is a valid name for a directory
 bucket created in
    //Region us-west-2, Availability Zone 2

    CreateBucketConfiguration bucketConfiguration =
 CreateBucketConfiguration.builder()
             .location(LocationInfo.builder()
                     .type(LocationType.AVAILABILITY_ZONE)
                     .name("usw2-az1").build()) //this must match the Region and
 Availability Zone in your bucket name
```

```
            .bucket(BucketInfo.builder()
                   .type(BucketType.DIRECTORY)
                   .dataRedundancy(DataRedundancy.SINGLE_AVAILABILITY_ZONE)
                   .build()).build();
    try {

            CreateBucketRequest bucketRequest =
 CreateBucketRequest.builder().bucket(bucketName).createBucketConfiguration(bucketConfigurat
            CreateBucketResponse response = s3Client.createBucket(bucketRequest);
            System.out.println(response);
    }

    catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

AWS SDK for JavaScript

Contoh ini menunjukkan cara membuat bucket direktori dengan menggunakan file AWS SDK for JavaScript.

Example

```
// file.mjs, run with Node.js v16 or higher
// To use with the preview build, place this in a folder
// inside the preview build directory, such as /aws-sdk-js-v3/workspace/

import { S3 } from "@aws-sdk/client-s3";

const region = "us-east-1";
const zone = "use1-az4";
const suffix = `${zone}--x-s3`;

const s3 = new S3({ region });

const bucketName = `...--${suffix}`;

const createResponse = await s3.createBucket(
    { Bucket: bucketName,
```

```
      CreateBucketConfiguration: {Location: {Type: "AvailabilityZone", Name: zone},
      Bucket: { Type: "Directory", DataRedundancy: "SingleAvailabilityZone" }}
    }
   );
```

AWS SDK for .NET

Contoh ini menunjukkan cara membuat bucket direktori dengan menggunakan file AWS SDK for .NET.

Example

```
using (var amazonS3Client = new AmazonS3Client())
{
    var putBucketResponse = await amazonS3Client.PutBucketAsync(new PutBucketRequest
    {

       BucketName = "DOC-EXAMPLE-BUCKET--usw2-az1--x-s3",
       PutBucketConfiguration = new PutBucketConfiguration
       {
         BucketInfo = new BucketInfo { DataRedundancy =
 DataRedundancy.SingleAvailabilityZone, Type = BucketType.Directory },
         Location = new LocationInfo { Name = "usw2-az1", Type =
 LocationType.AvailabilityZone }
       }
     }).ConfigureAwait(false);
}
```

SDK for PHP

Contoh ini menunjukkan cara membuat bucket direktori dengan menggunakan file AWS SDK for PHP.

Example

```
require 'vendor/autoload.php';

$s3Client = new S3Client([

    'region'      => 'us-east-1',
]);
```

```
$result = $s3Client->createBucket([
    'Bucket' => 'doc-example-bucket--use1-az4--x-s3',
    'CreateBucketConfiguration' => [
        'Location' => ['Name'=> 'use1-az4', 'Type'=> 'AvailabilityZone'],
        'Bucket' => ["DataRedundancy" => "SingleAvailabilityZone" ,"Type" =>
 "Directory"]   ],
]);
```

SDK for Python

Contoh ini menunjukkan cara membuat bucket direktori dengan menggunakan file AWS SDK for Python (Boto3).

Example

```
import logging
import boto3
from botocore.exceptions import ClientError

def create_bucket(s3_client, bucket_name, availability_zone):
    '''
    Create a directory bucket in a specified Availability Zone

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket to create; for example, 'doc-example-bucket--usw2-
az1--x-s3'
    :param availability_zone: String; Availability Zone ID to create the bucket in,
 for example, 'usw2-az1'
    :return: True if bucket is created, else False
    '''

    try:
        bucket_config = {
                'Location': {
                    'Type': 'AvailabilityZone',
                    'Name': availability_zone
                },
                'Bucket': {
                    'Type': 'Directory',
                    'DataRedundancy': 'SingleAvailabilityZone'
                }
            }
        s3_client.create_bucket(
            Bucket = bucket_name,
```

```
            CreateBucketConfiguration = bucket_config
        )
    except ClientError as e:
        logging.error(e)
        return False
    return True


if __name__ == '__main__':
    bucket_name = 'BUCKET_NAME'
    region = 'us-west-2'
    availability_zone = 'usw2-az1'
    s3_client = boto3.client('s3', region_name = region)
    create_bucket(s3_client, bucket_name, availability_zone)
```

SDK for Ruby

Contoh ini menunjukkan cara membuat bucket direktori dengan menggunakan file AWS SDK for Ruby.

Example

```
s3 = Aws::S3::Client.new(region:'us-west-2')
s3.create_bucket(
  bucket: "bucket_base_name--az_id--x-s3",
  create_bucket_configuration: {
    location: { name: 'usw2-az1', type: 'AvailabilityZone' },
    bucket: { data_redundancy: 'SingleAvailabilityZone', type: 'Directory' }
  }
)
```

Menggunakan AWS CLI

Contoh ini menunjukkan cara membuat bucket direktori dengan menggunakan file AWS CLI. Untuk menggunakan perintah, ganti *placeholder input pengguna dengan informasi* Anda sendiri.

Saat membuat bucket direktori, Anda harus memberikan detail konfigurasi dan menggunakan konvensi penamaan berikut: *bucket-base-name*--*azid*--x-s3

```
aws s3api create-bucket
--bucket bucket-base-name--azid--x-s3
```

```
--create-bucket-configuration 'Location={Type=AvailabilityZone,Name=usw2-
az1},Bucket={DataRedundancy=SingleAvailabilityZone,Type=Directory}'
--region us-west-2
```

Untuk informasi lebih lanjut, lihat create-bucket di. AWS Command Line Interface

Melihat properti bucket direktori

Anda dapat melihat dan mengonfigurasi properti untuk bucket direktori Amazon S3 dengan menggunakan konsol Amazon S3. Lihat informasi yang lebih lengkap di Ember direktori dan Apa itu S3 Express One Zone?.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih tab Bucket direktori.
4. Di daftar Bucket direktori, pilih nama bucket yang ingin Anda lihat propertinya.
5. Pilih tab Properti.
6. Pada tab Properties, Anda dapat melihat properti berikut untuk bucket:

   - Ikhtisar bucket direktori — Anda dapat melihatWilayah AWS, Availability Zone, Amazon Resource Name (ARN), dan tanggal pembuatan bucket.
   - Enkripsi default – Amazon S3 menerapkan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi untuk semua bucket S3. Untuk bucket direktori, pengaturan ini tidak dapat dimodifikasi. Amazon S3 mengenkripsi sebuah objek sebelum menyimpannya ke disk, dan mendekripsi objek tersebut saat Anda mengunduhnya. Untuk informasi selengkapnya, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3.

     Untuk informasi selengkapnya tentang fitur untuk bucket direktori, lihat Fitur S3 Express One Zone.

Mengelola kebijakan bucket untuk bucket direktori

Anda dapat menambahkan, menghapus, memperbarui, dan melihat kebijakan bucket untuk bucket direktori Amazon S3 dengan menggunakan konsol Amazon S3 dan SDK. AWS Untuk informasi

selengkapnya, lihat topik berikut. Untuk informasi selengkapnya tentang tindakan dan kunci kondisi yang didukung AWS Identity and Access Management (IAM) untuk S3 Express One Zone, lihat. AWS Identity and Access Management (IAM) untuk S3 Express One Zone Untuk contoh kebijakan bucket untuk bucket direktori, lihat Contoh kebijakan bucket direktori untuk S3 Express One Zone.

Topik

- Menambahkan kebijakan bucket
- Melihat kebijakan bucket
- Menghapus kebijakan bucket

Menambahkan kebijakan bucket

Untuk menambahkan kebijakan bucket ke bucket direktori, Anda dapat menggunakan konsol Amazon S3, AWS SDK, atau. AWS CLI

Menggunakan konsol S3

Untuk membuat atau mengedit kebijakan bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih tab Bucket direktori.
4. Di daftar Bucket direktori, pilih nama bucket tempat Anda ingin mengunggah folder dan file.
5. Pilih tab Izin.
6. Di Bawah Kebijakan bucket, pilih Edit. Halaman Edit kebijakan bucket akan muncul.
7. Untuk membuat kebijakan secara otomatis, pilih Generator kebijakan.

   Jika Anda memilih Policy Generator, AWS Policy Generator akan terbuka di jendela baru.

   Jika Anda tidak ingin menggunakan Generator AWS Kebijakan, Anda dapat menambahkan atau mengedit pernyataan JSON di bagian Kebijakan.

   a. Pada halaman AWS Pembuat Kebijakan, untuk Pilih Jenis Kebijakan, pilih Kebijakan Bucket S3.
   b. Tambahkan pernyataan dengan memasukkan informasi di bidang yang disediakan, lalu pilih Tambah Pernyataan. Ulangi langkah ini untuk pernyataan sebanyak yang ingin Anda

tambahkan. Untuk informasi selengkapnya tentang persyaratan kebijakan, lihat Referensi kebijakan IAM JSON di Panduan Pengguna IAM.

> **Note**
>
> Demi kenyamanan Anda, halaman kebijakan Edit bucket menampilkan Bucket ARN (Nama Sumber Daya Amazon) dari bucket saat ini di atas bidang teks Kebijakan. Anda dapat menyalin ARN ini untuk digunakan dalam pernyataan di halaman Pembuat Kebijakan AWS .

c. Setelah Anda selesai menambahkan pernyataan, pilih Buat Kebijakan.

d. Salin teks kebijakan yang dihasilkan, pilih Tutup, dan kembali ke halaman Edit kebijakan bucket di konsol Amazon S3.

8. Di kotak Kebijakan, edit kebijakan yang ada atau tempel kebijakan bucket dari AWS Policy Generator. Pastikan peringatan keamanan, kesalahan, peringatan umum, dan saran telah ditangani sebelum menyimpan kebijakan.

> **Note**
>
> Kebijakan bucket dibatasi hingga ukuran 20 KB.

9. Pilih Simpan perubahan, yang mengembalikan Anda ke tab Izin.

Menggunakan AWS SDK

SDK for Java 2.x

Example

`PutBucketPolicy` AWS SDK for Java 2.x

```
public static void setBucketPolicy(S3Client s3Client, String bucketName, String
 policyText) {

       //sample policy text
       /**
        * policy_statement = {
         *         'Version': '2012-10-17',
         *         'Statement': [
```

```
        *             {
        *                 'Sid': 'AdminPolicy',
        *                 'Effect': 'Allow',
        *                 'Principal': {
        *                     "AWS": "111122223333"
        *                 },
        *                 'Action': 's3express:*',
        *                 'Resource':
 'arn:aws:s3express:region:111122223333:bucket/bucket-base-name--azid--x-s3'
        *             }
        *         ]
        *     }
        */
        System.out.println("Setting policy:");
        System.out.println("----");
        System.out.println(policyText);
        System.out.println("----");
        System.out.format("On Amazon S3 bucket: \"%s\"\n", bucketName);

        try {
            PutBucketPolicyRequest policyReq = PutBucketPolicyRequest.builder()
                    .bucket(bucketName)
                    .policy(policyText)
                    .build();
            s3Client.putBucketPolicy(policyReq);
            System.out.println("Done!");
        }

        catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

Menggunakan AWS CLI

Contoh ini menunjukkan cara menambahkan kebijakan bucket ke bucket direktori dengan menggunakan AWS CLI. Untuk menggunakan perintah, ganti *placeholder input pengguna dengan informasi* Anda sendiri.

```
aws s3api put-bucket-policy --bucket bucket-base-name--azid--x-s3 --policy file://
bucket_policy.json
```

bucket_policy.json:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AdminPolicy",
            "Effect": "Allow",
            "Principal": {
                "AWS": "111122223333"
            },
            "Action": "s3express*",
            "Resource": "arn:aws:s3express:us-west-2:111122223333:bucket/"
        }
    ]
}
```

Untuk informasi lebih lanjut, lihat put-bucket-policydi AWS Command Line Interface.

Melihat kebijakan bucket

Untuk melihat kebijakan bucket untuk bucket direktori, gunakan contoh berikut.

Menggunakan AWS CLI

Contoh ini menunjukkan cara melihat kebijakan bucket yang dilampirkan ke bucket direktori dengan menggunakan AWS CLI. Untuk menggunakan perintah, ganti *placeholder input pengguna dengan informasi* Anda sendiri.

```
aws s3api get-bucket-policy --bucket bucket-base-name--azid--x-s3
```

Untuk informasi lebih lanjut, lihat get-bucket-policydi AWS Command Line Interface.

Menghapus kebijakan bucket

Untuk menghapus kebijakan bucket untuk bucket direktori, gunakan contoh berikut.

Menggunakan AWS SDK

SDK for Java 2.x

Example

`DeleteBucketPolicy` AWS SDK for Java 2.x

```
public static void deleteBucketPolicy(S3Client s3Client, String bucketName) {
      try {
          DeleteBucketPolicyRequest deleteBucketPolicyRequest =
 DeleteBucketPolicyRequest
                  .builder()
                  .bucket(bucketName)
                  .build()
          s3Client.deleteBucketPolicy(deleteBucketPolicyRequest);
          System.out.println("Successfully deleted bucket policy");
      }

      catch (S3Exception e) {
          System.err.println(e.awsErrorDetails().errorMessage());
          System.exit(1);
      }
```

Menggunakan AWS CLI

Contoh ini menunjukkan cara menghapus kebijakan bucket untuk bucket direktori dengan menggunakan AWS CLI. Untuk menggunakan perintah, ganti *placeholder input pengguna dengan informasi* Anda sendiri.

```
aws s3api delete-bucket-policy --bucket bucket-base-name--azid--x-s3
```

Untuk informasi lebih lanjut, lihat delete-bucket-policydi AWS Command Line Interface.

Mengosongkan bucket direktori

Anda dapat mengosongkan bucket direktori Amazon S3 dengan menggunakan konsol Amazon S3. Untuk informasi selengkapnya tentang bucket direktori, lihat Ember direktori.

Sebelum Anda mengosongkan bucket direktori, perhatikan hal berikut ini:

- Saat mengosongkan bucket direktori, Anda menghapus semua objek tetapi menyimpan bucket direktori.
- Setelah Anda mengosongkan bucket direktori, tindakan kosong tidak dapat dibatalkan.
- Objek yang ditambahkan ke bucket direktori saat aksi bucket kosong sedang berlangsung mungkin akan dihapus.

Jika Anda juga ingin menghapus ember, perhatikan hal berikut:

- Semua objek di bucket direktori harus dihapus sebelum bucket itu sendiri dapat dihapus.
- Unggahan multibagian yang sedang berlangsung di bucket direktori harus dibatalkan sebelum bucket itu sendiri dapat dihapus.

> **Note**
>
> `s3 rm`Perintah melalui AWS Command Line Interface (CLI), `delete` operasi melalui Mountpoint, dan tombol opsi Empty bucket melalui tombol tidak dapat menghapus unggahan multibagian AWS Management Console yang sedang berlangsung di bucket direktori. Untuk menghapus unggahan multibagian yang sedang berlangsung ini, gunakan `ListMultipartUploads` operasi untuk mencantumkan unggahan multibagian yang sedang berlangsung di bucket dan gunakan `AbortMultupartUpload` operasi untuk membatalkan semua unggahan multibagian yang sedang berlangsung.

Untuk mengosongkan bucket direktori, lihat Menghapus bucket direktori. Untuk membatalkan unggahan multipart yang sedang berlangsung, lihat. the section called “Membatalkan unggahan multibagian”

Untuk menghapus bucket bertujuan umum, lihat Mengosongkan bucket.

Menggunakan konsol S3

Untuk mengosongkan ember direktori

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih tab Bucket direktori.

4. Pilih tombol opsi di sebelah nama ember yang ingin Anda kosongkan, lalu pilih Kosong.
5. Di halaman Bucket kosong, konfirmasikan bahwa Anda ingin mengosongkan bucket dengan mengetik **`permanently delete`** ke dalam bidang teks, lalu pilih Kosongkan.
6. Pantau perkembangan proses pengosongan bucket di halaman Empty bucket: status.

Menghapus bucket direktori

Anda hanya dapat menghapus ember direktori Amazon S3 kosong. Sebelum menghapus bucket direktori, Anda harus menghapus semua objek di bucket dan membatalkan semua unggahan multipart yang sedang berlangsung.

Untuk mengosongkan bucket direktori, lihat Mengosongkan bucket direktori. Untuk membatalkan unggahan multipart yang sedang berlangsung, lihat. the section called “Membatalkan unggahan multibagian”

Untuk menghapus bucket bertujuan umum, lihat Menghapus bucket.

Menggunakan konsol S3

Setelah mengosongkan bucket direktori dan membatalkan semua unggahan multipart yang sedang berlangsung, Anda dapat menghapus bucket.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih tab Bucket direktori.
4. Dalam daftar bucket Direktori, pilih tombol opsi di sebelah bucket yang ingin Anda hapus.
5. Pilih Hapus.
6. Pada halaman Delete bucket, masukkan nama bucket di kolom teks untuk mengonfirmasi penghapusan bucket Anda.

   > **Important**
   >
   > Menghapus bucket direktori tidak dapat dibatalkan.

7. Untuk menghapus bucket direktori, pilih Hapus bucket.

Menggunakan AWS SDK

Contoh berikut menghapus bucket direktori dengan menggunakan AWS SDK for Java 2.x dan AWS SDK for Python (Boto3).

SDK for Java 2.x

Example

```
public static void deleteBucket(S3Client s3Client, String bucketName) {

    try {
        DeleteBucketRequest del = DeleteBucketRequest.builder()
                .bucket(bucketName)
                .build();
        s3Client.deleteBucket(del);
        System.out.println("Bucket " + bucketName + " has been deleted");
    }
    catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

SDK for Python

Example

```
import logging
import boto3
from botocore.exceptions import ClientError

def delete_bucket(s3_client, bucket_name):
    '''
    Delete a directory bucket in a specified Region

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket to delete; for example, 'doc-example-bucket--usw2-
az1--x-s3'
    :return: True if bucket is deleted, else False
    '''
```

```
    try:
        s3_client.delete_bucket(Bucket = bucket_name)
    except ClientError as e:
        logging.error(e)
        return False
    return True

if __name__ == '__main__':
    bucket_name = 'BUCKET_NAME'
    region = 'us-west-2'
    s3_client = boto3.client('s3', region_name = region)
```

Menggunakan AWS CLI

Contoh ini menunjukkan cara menghapus bucket direktori dengan menggunakan file AWS CLI. Untuk menggunakan perintah ganti *placeholder input pengguna dengan informasi* Anda sendiri.

```
aws s3api delete-bucket --bucket bucket-base-name--azid--x-s3 --region us-west-2
```

Untuk informasi selengkapnya, lihat delete-bucket di. AWS Command Line Interface

Mendaftar bucket direktori

Contoh berikut menunjukkan cara membuat daftar bucket direktori dengan menggunakan AWS SDK dan CLI AWS .

Menggunakan AWS SDK

SDK for Java 2.x

Example

Contoh berikut mencantumkan bucket direktori dengan menggunakan file. AWS SDK for Java 2.x

```
public static void listBuckets(S3Client s3Client) {
       try {
           ListDirectoryBucketsRequest listDirectoryBucketsRequest =
ListDirectoryBucketsRequest.builder().build();
           ListDirectoryBucketsResponse response =
s3Client.listDirectoryBuckets(listDirectoryBucketsRequest);
```

```
            if (response.hasBuckets()) {
                for (Bucket bucket: response.buckets()) {
                    System.out.println(bucket.name());
                    System.out.println(bucket.creationDate());
                }
            }
        }

        catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

SDK for Python

Example

Contoh berikut mencantumkan bucket direktori dengan menggunakan file. AWS SDK for Python (Boto3)

```
import logging
import boto3
from botocore.exceptions import ClientError

def list_directory_buckets(s3_client):
'''
Prints a list of all directory buckets in a Region

:param s3_client: boto3 S3 client
:return: True if there are buckets in the Region, else False
'''
try:
    response = s3_client.list_directory_buckets()
    for bucket in response['Buckets']:
        print (bucket['Name'])
except ClientError as e:
    logging.error(e)
    return False
return True
```

```
if __name__ == '__main__':
    region = 'us-east-1'
    s3_client = boto3.client('s3', region_name = region)
    list_directory_buckets(s3_client)
```

AWS SDK for .NET

Example

Contoh berikut mencantumkan bucket direktori dengan menggunakan file. AWS SDK for .NET

```
var listDirectoryBuckets = await amazonS3Client.ListDirectoryBucketsAsync(new
 ListDirectoryBucketsRequest
{
  MaxDirectoryBuckets = 10
  }).ConfigureAwait(false);
```

SDK for PHP

Example

Contoh berikut mencantumkan bucket direktori dengan menggunakan file. AWS SDK for PHP

```
require 'vendor/autoload.php';

$s3Client = new S3Client([
    'region'      => 'us-east-1',
]);
$result = $s3Client->listDirectoryBuckets();
```

SDK for Ruby

Example

Contoh berikut mencantumkan bucket direktori dengan menggunakan file. AWS SDK for Ruby

```
s3 = Aws::S3::Client.new(region:'us-west-1')
```

```
s3.list_directory_buckets
```

Menggunakan AWS CLI

`list-directory-buckets`Contoh perintah berikut menunjukkan bagaimana Anda dapat menggunakan AWS CLI untuk daftar bucket direktori Anda di wilayah *`us-east-1`*. Untuk menjalankan perintah ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3api list-directory-buckets --region us-east-1
```

Untuk informasi selengkapnya, lihat list-directory-buckets dalam AWS CLI Referensi Perintah.

Menggunakan **`HeadBucket`** dengan ember direktori

Contoh AWS SDK berikut menunjukkan cara menggunakan operasi `HeadBucket` API untuk menentukan apakah bucket direktori Amazon S3 ada dan apakah Anda memiliki izin untuk mengaksesnya.

Menggunakan AWS SDK

AWS SDK for Java 2.x Contoh berikut menunjukkan cara menentukan apakah ada bucket dan apakah Anda memiliki izin untuk mengaksesnya.

SDK for Java 2.x

Example

AWS SDK for Java 2.x

```
public static void headBucket(S3Client s3Client, String bucketName) {
   try {
        HeadBucketRequest headBucketRequest = HeadBucketRequest
                .builder()
                .bucket(bucketName)
                .build();
        s3Client.headBucket(headBucketRequest);
        System.out.format("Amazon S3 bucket: \"%s\" found.", bucketName);
   }
```

```
    catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

Menggunakan AWS CLI

Perintah `head-bucket` contoh berikut menunjukkan bagaimana Anda dapat menggunakan AWS CLI untuk menentukan apakah ada bucket direktori dan apakah Anda memiliki izin untuk mengaksesnya. Untuk menjalankan perintah ini, ganti placeholder input pengguna dengan informasi Anda sendiri.

```
aws s3api head-bucket --bucket bucket-base-name--azid--x-s3
```

Untuk informasi selengkapnya, lihat head-bucket dalam AWS CLI Referensi Perintah.

Bekerja dengan objek dalam ember direktori

Setelah membuat bucket direktori Amazon S3, Anda dapat bekerja dengan objek menggunakan konsol Amazon S3 AWS Command Line Interface ,AWS CLI(), dan SDK. AWS

Untuk informasi selengkapnya tentang operasi objek massal dengan objek yang disimpan di kelas penyimpanan S3 Express One Zone, lihatManajemen objek. Untuk informasi selengkapnya tentang mengimpor, mengunggah, menyalin, menghapus, dan mengunduh objek dan membaca metadata dari objek di bucket direktori, lihat topik berikut.

Topik

- Mengimpor objek ke dalam bucket direktori
- Menggunakan Operasi Batch dengan S3 Express One Zone
- Mengunggah objek ke bucket direktori
- Menggunakan unggahan multibagian dengan bucket direktori
- Menyalin objek ke bucket direktori
- Menghapus objek dalam bucket direktori
- Mengunduh objek dalam ember direktori
- Menggunakan HeadObject dengan ember direktori

Mengimpor objek ke dalam bucket direktori

Setelah membuat bucket direktori di Amazon S3, Anda dapat mengisi bucket baru dengan data menggunakan tindakan impor. Impor adalah metode yang disederhanakan untuk membuat tugas Operasi Batch S3 guna menyalin objek dari bucket bertujuan umum ke bucket direktori.

> **Note**
>
> Batas berikut berlaku untuk tugas impor:
>
> - Bucket sumber dan bucket tujuan harus berada di Wilayah AWS dan akun yang sama.
> - Bucket sumber tidak dapat berupa bucket direktori.
> - Objek yang lebih besar dari 5GB tidak didukung dan akan dihilangkan dari operasi salin.
> - Objek di kelas penyimpanan Glacier Flexible Retrieval, Glacier Deep Archive, Intelligent-Tiering Archive Access, dan Intelligent-Tiering Deep Archive harus dipulihkan sebelum dapat diimpor.
> - Objek yang diimpor dengan algoritma checksum MD5 dikonversi untuk menggunakan checksum CRC32.
> - Objek yang diimpor menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3).
> - Objek yang diimpor menggunakan kelas penyimpanan Express One Zone, yang memiliki struktur harga berbeda dari kelas penyimpanan yang digunakan oleh bucket tujuan umum. Pertimbangkan perbedaan biaya ini saat mengimpor banyak objek.

Saat mengonfigurasi tugas impor, Anda menentukan bucket sumber atau awalan tempat objek yang ada akan disalin. Anda juga menyediakan peran (IAM) AWS Identity and Access Management yang memiliki izin untuk mengakses objek sumber. Amazon S3 kemudian memulai tugas Operasi Batch yang menyalin objek dan secara otomatis menerapkan kelas penyimpanan dan pengaturan checksum yang sesuai.

Untuk mengonfigurasi tugas impor, Anda menggunakan konsol Amazon S3.

Menggunakan konsol Amazon S3

Mengimpor objek ke dalam bucket direktori

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket, lalu pilih tab bucket Direktori. Pilih tombol opsi di samping bucket direktori yang ingin Anda tempati untuk mengimpor objek.
3. Pilih Impor.
4. Untuk Sumber, masukkan bucket bertujuan umum (atau jalur bucket termasuk awalan) yang berisi objek yang ingin Anda impor. Untuk memilih bucket bertujuan umum yang ada dari daftar, pilih Jelajahi S3.
5. Untuk Izin mengakses dan menyalin objek sumber, lakukan salah satu hal berikut untuk menentukan peran IAM dengan izin yang diperlukan untuk mengimpor objek sumber Anda:

   - Untuk mengizinkan Amazon S3 membuat peran IAM baru atas nama Anda, pilih Buat peran IAM baru.

     > **Note**
     >
     > Jika objek sumber Anda dienkripsi dengan enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS), jangan pilih opsi Buat peran IAM baru. Sebagai gantinya, tentukan peran IAM lama yang memiliki izin `kms:Decrypt`. Amazon S3 akan menggunakan izin ini untuk mendekripsi objek Anda. Selama proses impor, Amazon S3 kemudian akan mengenkripsi ulang objek tersebut dengan menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3).

   - Untuk memilih peran IAM yang ada dari daftar, pilih Pilih dari peran IAM yang ada.
   - Untuk menentukan peran IAM yang ada dengan memasukkan Amazon Resource Name (ARN), pilih Masukkan ARN peran IAM, lalu masukkan ARN di bidang yang sesuai.
6. Tinjau informasi yang ditampilkan di bagian Tujuan dan Pengaturan objek yang disalin. Jika informasi di bagian Tujuan sudah benar, pilih Impor untuk memulai tugas penyalinan.

   Konsol Amazon S3 menampilkan status tugas baru Anda di halaman Operasi Batch. Untuk informasi selengkapnya tentang tugas, pilih tombol opsi di sebelah nama tugas, lalu pada menu Tindakan, pilih Lihat detail. Untuk membuka bucket direktori tempat objek akan diimpor, pilih Lihat tujuan impor.

Menggunakan Operasi Batch dengan S3 Express One Zone

Anda dapat menggunakan Operasi Batch Amazon S3 untuk melakukan operasi pada objek yang disimpan dalam bucket S3. Untuk mempelajari Operasi Batch S3, lihat Melakukan operasi batch skala besar pada objek Amazon S3.

Topik berikut membahas melakukan operasi batch pada objek yang disimpan di kelas penyimpanan S3 Express One Zone dalam bucket direktori.

Topik

- Menggunakan Operasi Batch dengan bucket direktori
- Perbedaan utama

Menggunakan Operasi Batch dengan bucket direktori

Anda dapat melakukan operasi Salin dan operasi AWS Lambdafungsi Invoke pada objek yang disimpan dalam bucket direktori. Dengan Copy, Anda dapat menyalin objek di antara bucket dengan tipe yang sama (misalnya, dari bucket direktori ke bucket direktori). Anda juga dapat menyalin antara bucket tujuan umum dan bucket direktori. Dengan Fungsi AWS Lambda invokasi, Anda dapat menggunakan fungsi Lambda untuk melakukan tindakan pada objek di bucket direktori Anda dengan kode yang Anda tentukan.

Menyalin objek

Anda dapat menyalin antara jenis bucket yang sama atau antara bucket direktori dan bucket tujuan umum. Saat menyalin ke bucket direktori, Anda harus menggunakan format Amazon Resource Name (ARN) yang benar untuk jenis bucket ini. Format ARN untuk bucket direktori adalah `arn:aws:s3express:region:account-id:bucket/bucket-base-name--x-s3`.

Anda juga dapat mengisi bucket direktori Anda dengan data menggunakan tindakan Impor di konsol S3. Impor adalah metode yang disederhanakan untuk membuat pekerjaan Operasi Batch guna menyalin objek dari bucket tujuan umum ke bucket direktori. Untuk tugas Impor salinan dari bucket tujuan umum ke bucket direktori, S3 secara otomatis menghasilkan manifes. Untuk informasi selengkapnya, lihat Mengimpor objek ke bucket direktori dan Menentukan manifes.

Memanggil fungsi Lambda () **`LambdaInvoke`**

Ada persyaratan khusus untuk menggunakan Operasi Batch untuk menginvokasi fungsi Lambda yang bertindak pada bucket direktori. Misalnya, Anda harus menyusun permintaan Lambda Anda

dengan menggunakan skema pemanggilan v2 JSON, dan menentukan InvocationSchemaVersion 2.0 kapan Anda membuat pekerjaan. Untuk informasi selengkapnya, lihat Memanggil AWS Lambda fungsi.

Perbedaan utama

Berikut ini adalah daftar perbedaan utama saat Anda menggunakan Operasi Batch untuk melakukan operasi massal pada objek yang disimpan dalam bucket direktori dengan kelas penyimpanan S3 Express One Zone:

- Amazon S3 secara otomatis mengenkripsi semua objek baru yang diunggah ke bucket S3. Konfigurasi enkripsi default bucket S3 selalu diaktifkan dan diatur secara minimum ke enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3). Untuk bucket direktori, hanya SSSE-S3 yang didukung. Jika Anda membuat `CopyObject` permintaan yang menetapkan enkripsi sisi server dengan kunci yang disediakan pelanggan (SSE-C) atau enkripsi sisi server dengan () kunci AWS Key Management Service (SSE-KMSAWS KMS) pada bucket direktori (sumber atau tujuan), respons akan mengembalikan kesalahan HTTP. `400 (Bad Request)`
- Objek dalam bucket direktori tidak dapat ditandai. Anda hanya dapat menentukan set tanda kosong. Secara default, Operasi Batch menyalin tanda. Jika Anda menyalin objek yang memiliki tag antara bucket tujuan umum dan bucket direktori, Anda akan menerima respons. `501 (Not Implemented)`
- S3 Express One Zone menawarkan Anda opsi untuk memilih algoritma checksum yang digunakan untuk memvalidasi data Anda selama unggahan atau unduhan. Anda dapat memilih salah satu algoritma pemeriksaan integritas data Secure Hash Algorithms (SHA) atau Cyclic Redundancy Check (CRC) berikut: CRC32, CRC32C, SHA-1, dan SHA-256. Checksum berbasis MD5 tidak didukung dengan kelas penyimpanan S3 Express One Zone.
- Secara default, semua bucket Amazon S3 menyetel setelan Kepemilikan Objek S3 ke pemilik bucket yang diberlakukan dan daftar kontrol akses (ACL) dinonaktifkan. Untuk bucket direktori, pengaturan ini tidak dapat dimodifikasi. Anda dapat menyalin objek dari bucket tujuan umum ke bucket direktori. Namun, Anda tidak dapat menimpa ACL default saat menyalin ke atau dari bucket direktori.
- Terlepas dari bagaimana Anda menentukan manifes Anda, daftar itu sendiri harus disimpan dalam bucket tujuan umum. Operasi Batch tidak dapat mengimpor manifes yang ada dari (atau menyimpan manifes yang dihasilkan ke) bucket direktori. Namun, objek yang dijelaskan dalam manifes dapat disimpan dalam bucket direktori.
- Operasi Batch tidak dapat menentukan bucket direktori sebagai lokasi dalam laporan Inventaris S3. Laporan inventaris tidak mendukung bucket direktori. Anda dapat membuat file manifes untuk objek

dalam bucket direktori dengan menggunakan operasi `ListObjectsV2` API untuk mencantumkan objek. Anda kemudian dapat memasukkan daftar dalam file CSV.

Memberi Akses

Untuk melakukan tugas penyalinan, Anda harus memiliki izin berikut:

- Untuk menyalin objek dari satu bucket direktori ke bucket direktori lain, Anda harus memiliki izin `s3express:CreateSession`.
- Untuk menyalin objek dari bucket direktori ke bucket tujuan umum, Anda harus memiliki izin `s3express:CreateSession` dan izin `s3:PutObject`untuk menulis salinan objek ke bucket tujuan.
- Untuk menyalin objek dari bucket tujuan umum ke bucket direktori, Anda harus memiliki `s3express:CreateSession` izin dan `s3:GetObject` izin untuk membaca objek sumber yang sedang disalin.

  Untuk informasi selengkapnya, lihat CopyObject dalam Referensi API Amazon Simple Storage Service.
- Untuk menginvokasi fungsi Lambda, Anda harus memberikan izin ke sumber daya berdasarkan fungsi Lambda Anda. Untuk menentukan izin mana yang diperlukan, periksa operasi API yang sesuai.

Mengunggah objek ke bucket direktori

Setelah membuat bucket direktori Amazon S3, Anda dapat mengunggah objek ke dalamnya. Contoh berikut menunjukkan cara mengunggah objek ke bucket direktori menggunakan konsol S3 dan AWS SDK. Untuk informasi tentang operasi pengunggahan objek massal dengan S3 Express One Zone, lihatManajemen objek.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih tab Bucket direktori.
4. Pilih nama bucket tempat Anda ingin mengunggah folder atau file Anda.

5. Dalam daftar Objek, pilih Unggah.
6. Pada halaman Unggah, lakukan salah satu hal berikut:
   - Seret dan lepas file dan folder ke area unggahan bertitik.
   - Pilih Tambahkan file atau Tambah folder, pilih file atau folder yang akan diunggah, lalu pilih Buka atau Unggah.
7. Di bawah Checksum, pilih fungsi Checksum yang ingin Anda gunakan.

   (Opsional) Jika Anda mengunggah satu objek yang berukuran kurang dari 16 MB, Anda juga dapat menentukan nilai checksum yang telah dihitung sebelumnya. Saat Anda memberikan nilai yang telah dihitung sebelumnya, Amazon S3 membandingkannya dengan nilai yang dihitung dengan menggunakan fungsi checksum yang dipilih. Jika nilai tidak cocok, unggahan tidak akan dimulai.
8. Opsi di bagian Izin dan Properti secara otomatis diatur ke pengaturan default dan tidak dapat dimodifikasi. Blokir Akses Publik diaktifkan secara otomatis, dan Versi S3 dan Kunci Objek S3 tidak dapat diaktifkan untuk bucket direktori.

   (Opsional) Jika Anda ingin menambahkan metadata dalam pasangan nilai kunci ke objek Anda, perluas bagian Properti, lalu di bagian Metadata, pilih Tambahkan metadata.
9. Untuk mengunggah file dan folder yang terdaftar, pilih Unggah.

   Amazon S3 mengunggah objek dan folder Anda. Setelah unggahan selesai, Anda melihat pesan sukses di halaman Unggahan: status.

Menggunakan AWS SDK

SDK for Java 2.x

Example

```
public static void putObject(S3Client s3Client, String bucketName, String objectKey,
 Path filePath) {
       //Using File Path to avoid loading the whole file into memory
       try {
           PutObjectRequest putObj = PutObjectRequest.builder()
                   .bucket(bucketName)
                   .key(objectKey)
                   //.metadata(metadata)
                   .build();
```

```
            s3Client.putObject(putObj, filePath);
            System.out.println("Successfully placed " + objectKey +" into bucket
 "+bucketName);

        }

        catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
}
```

SDK for Python

Example

```
import boto3
import botocore
from botocore.exceptions import ClientError


def put_object(s3_client, bucket_name, key_name, object_bytes):
    """
    Upload data to a directory bucket.
    :param s3_client: The boto3 S3 client
    :param bucket_name: The bucket that will contain the object
    :param key_name: The key of the object to be uploaded
    :param object_bytes: The data to upload
    """
    try:
        response = s3_client.put_object(Bucket=bucket_name, Key=key_name,
                                        Body=object_bytes)
        print(f"Upload object '{key_name}' to bucket '{bucket_name}'.")
        return response
    except ClientError:
        print(f"Couldn't upload object '{key_name}' to bucket '{bucket_name}'.")
        raise

def main():
    # Share the client session with functions and objects to benefit from S3 Express
 One Zone auth key
    s3_client = boto3.client('s3')
    # Directory bucket name must end with --azid--x-s3
```

```
    resp = put_object(s3_client, 'doc-bucket-example--use1-az5--x-s3', 'sample.txt',
 b'Hello, World!')
    print(resp)

if __name__ == "__main__":
    main()
```

Menggunakan AWS CLI

`put-object`Contoh perintah berikut menunjukkan bagaimana Anda dapat menggunakan AWS CLI untuk meng-upload objek dari Amazon S3. Untuk menjalankan perintah ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3api put-object --bucket bucket-base-name--azid--x-s3 --key sampleinut/file001.bin
 --body bucket-seed/file001.bin
```

Untuk informasi selengkapnya, lihat put-object dalam AWS CLI Referensi Perintah.

Menggunakan unggahan multibagian dengan bucket direktori

Anda dapat menggunakan proses upload multipart untuk mengunggah satu objek sebagai satu set bagian. Setiap bagian merupakan bagian data objek yang saling berkaitan. Anda dapat mengunggah bagian-bagian objek tersebut secara independen dan dengan urutan apa pun. Jika ada transmisi bagian mana pun yang gagal, Anda dapat mentransmisikan ulang bagian tersebut tanpa memengaruhi bagian lainnya. Setelah semua bagian objek Anda diunggah, Amazon S3 merakit bagian-bagian ini dan menciptakan objek. Secara umum, saat ukuran objek Anda mencapai 100 MB, Anda harus mempertimbangkan untuk menggunakan unggahan multibagian daripada mengunggah objek tersebut dalam satu operasi.

Penggunaan unggahan multibagian memberikan keuntungan sebagai berikut:

- Peningkatan throughput–Anda dapat mengunggah bagian-bagian secara paralel untuk meningkatkan throughput.
- Pemulihan cepat dari masalah jaringan apa pun - Ukuran bagian yang lebih kecil meminimalkan dampak memulai ulang unggahan yang gagal karena kesalahan jaringan.
- Jeda dan pelanjutan pengunggahan objek–Anda dapat mengunggah bagian-bagian objek kapan saja. Setelah Anda memulai unggahan multipart, tidak ada tanggal kedaluwarsa. Anda harus secara eksplisit menyelesaikan atau membatalkan unggahan multipart.

- Mulai unggahan sebelum Anda mengetahui ukuran akhir objek–Anda dapat mengunggah sebuah objek selagi Anda membuatnya.

Kami menyarankan Anda menggunakan unggahan multipart dengan cara berikut:

- Jika Anda mengunggah objek besar melalui jaringan bandwidth tinggi yang stabil, gunakan unggahan multipart untuk memaksimalkan penggunaan bandwidth yang tersedia dengan mengunggah bagian objek secara paralel untuk kinerja multi-threaded.
- Jika Anda mengunggah melalui jaringan jerawatan, gunakan unggahan multibagian untuk meningkatkan ketahanan terhadap kesalahan jaringan dengan menghindari pengunggahan dimulai ulang. Saat menggunakan unggahan multibagian, Anda perlu mencoba mengunggah kembali hanya bagian yang terputus selama pengunggahan. Anda tidak perlu mengunggah ulang objek Anda dari awal.

Saat Anda menggunakan unggahan multibagian untuk mengunggah objek ke kelas penyimpanan Amazon S3 Express One Zone dalam bucket direktori, proses pengunggahan multibagian mirip dengan proses menggunakan unggahan multibagian untuk mengunggah objek ke bucket tujuan umum. Namun, ada beberapa perbedaan penting.

Untuk informasi selengkapnya tentang penggunaan unggahan multibagian untuk mengunggah objek ke S3 Express One Zone, lihat topik berikut.

Topik

- Proses pengunggahan multipart
- Checksum dengan operasi unggahan multibagian
- Operasi pengunggahan multibagian serentak
- Unggahan dan harga multipart
- Operasi dan izin API unggahan multibagian
- Contoh

Proses pengunggahan multipart

Unggahan multipart adalah proses tiga langkah:

- Anda memulai unggahan.
- Anda mengunggah bagian objek.

- Setelah Anda mengunggah semua bagian, Anda menyelesaikan unggahan multibagian.

Setelah menerima permintaan upload multipart lengkap, Amazon S3 membuat objek dari bagian yang diunggah, dan Anda kemudian dapat mengakses objek seperti halnya objek lain di bucket Anda.

Menginisiasi unggahan multibagian

Saat Anda mengirim permintaan untuk memulai unggahan multibagian, Amazon S3 mengirimkan respons dengan ID unggahan, yang merupakan pengidentifikasi unik untuk unggahan multibagian Anda. Anda harus menyertakan ID unggahan ini setiap kali Anda mengunggah bagian, mendaftar bagian, menyelesaikan unggahan, atau membatalkan sebuah unggahan.

Pengunggahan bagian-bagian

Saat mengunggah sebuah bagian, selain ID pengunggahan, Anda harus menentukan nomor bagiannya. Saat Anda menggunakan unggahan multibagian dengan S3 Express One Zone, nomor bagian multipart harus berupa nomor bagian yang berurutan. Jika Anda mencoba menyelesaikan permintaan unggahan multibagian dengan nomor bagian yang tidak berurutan, kesalahan `HTTP 400 Bad Request` (Pesanan Bagian Tidak Valid) akan dihasilkan.

Nomor bagian secara unik mengidentifikasi bagian dan posisinya dalam objek yang Anda unggah. Jika Anda mengunggah bagian baru dengan menggunakan nomor bagian yang sama dengan bagian yang diunggah sebelumnya, bagian yang diunggah sebelumnya akan ditimpa.

Kapan pun Anda mengunggah sebuah bagian, Amazon S3 akan menampilkan header tag entitas (ETag) dalam responsnya. Untuk setiap unggahan bagian, Anda harus mencatat nomor bagian dan nilai ETag. Nilai ETag untuk semua unggahan bagian objek akan tetap sama, tetapi setiap bagian akan diberi nomor bagian yang berbeda. Anda harus memasukkan nilai-nilai ini dalam permintaan selanjutnya untuk menyelesaikan unggahan multibagian.

Amazon S3 secara otomatis mengenkripsi semua objek baru yang diunggah ke bucket S3. Saat melakukan pengunggahan multibagian, jika Anda tidak menentukan informasi enkripsi dalam permintaan, pengaturan enkripsi bagian yang diunggah diatur ke konfigurasi enkripsi default bucket tujuan. Konfigurasi enkripsi default bucket Amazon S3 selalu diaktifkan dan diatur secara minimum ke enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3). Untuk bucket direktori, hanya SSE-S3 yang didukung. Untuk informasi selengkapnya, lihat Enkripsi Sisi Server dengan kunci terkelola Amazon S3 (SSE-S3).

Penyelesaian unggahan multibagian

Saat Anda menyelesaikan unggahan multibagian, Amazon S3 membuat objek dengan menggabungkan bagian-bagian dalam urutan menaik berdasarkan nomor bagian. Setelah permintaan selesai sepenuhnya, bagiantersebut tidak akan ada lagi.

Permintaan unggahan multibagian lengkap Anda harus menyertakan ID unggahan dan daftar nomor bagian dan nilai ETag yang sesuai. Respons Amazon S3 mencakup sebuah ETag yang secara unik mengidentifikasi data objek gabungan. ETag ini bukanlah hash MD5 dari data objek.

Pendaftaran unggahan multibagian

Anda dapat mendaftar bagian-bagian dari unggahan multibagian tertentu atau semua unggahan multibagian yang sedang berlangsung. Operasi daftar bagian menampilkan informasi bagian yang telah Anda unggah untuk unggahan multibagian tertentu. Untuk setiap permintaan daftar bagian, Amazon S3 akan menampilkan informasi bagian untuk unggahan multibagian tertentu, hingga maksimum 1.000 bagian. Jika ada lebih dari 1.000 bagian dalam unggahan multibagian, Anda harus menggunakan penomoran halaman untuk mengambil semua bagian.

Daftar suku cadang yang dikembalikan tidak termasuk bagian yang belum selesai diunggah. Dengan menggunakan operasi daftar unggahan multibagian, Anda dapat memperoleh daftar unggahan multibagian yang sedang berlangsung.

Unggahan multibagian yang sedang berlangsung adalah unggahan yang telah Anda mulai, tetapi belum selesai atau dibatalkan. Setiap permintaan akan ditampilkan sebanyak maksimum 1.000 unggahan multibagian. Jika ada lebih dari 1.000 unggahan multibagian yang sedang berlangsung, Anda harus mengirim permintaan tambahan untuk mengambil unggahan multibagian yang tersisa. Gunakan pendaftaran yang ditampilkan untuk verifikasi. Jangan menggunakan hasil pendaftaran ini saat mengirim permintaan selesaikan unggahan multibagian. Sebaliknya, simpan daftar nomor bagian Anda sendiri yang Anda tentukan saat mengunggah bagian dan nilai ETag yang ditampilkan oleh Amazon S3.

Untuk informasi selengkapnya tentang daftar unggahan multibagian, lihat ListPartsdi Referensi API Amazon Simple Storage Service.

Checksum dengan operasi unggahan multibagian

Saat Anda mengunggah objek, Anda dapat menentukan algoritma checksum untuk memeriksa integritas objek. MD5 tidak didukung untuk bucket direktori. Anda dapat menentukan salah satu

algoritma pemeriksaan integritas data Secure Hash Algorithms (SHA) atau Cyclic Redundancy Check (CRC) berikut:

- CRC32
- CRC32C
- SHA-1
- SHA-256

Anda dapat menggunakan Amazon S3 REST API atau AWS SDK untuk mengambil nilai checksum untuk masing-masing bagian dengan menggunakan atau. `GetObject` `HeadObject` Jika Anda ingin mengambil nilai checksum untuk masing-masing bagian dari unggahan multibagian yang masih dalam proses, Anda dapat menggunakan `ListParts`.

> **Important**
>
> Saat menggunakan algoritma checksum sebelumnya, nomor bagian multipart harus menggunakan nomor bagian berurutan. Jika Anda mencoba menyelesaikan permintaan unggahan multibagian dengan nomor bagian yang tidak berurutan, Amazon S3 menghasilkan kesalahan `HTTP 400 Bad Request` (Pesanan Bagian Tidak Valid).

Untuk informasi selengkapnya tentang cara kerja checksum dengan objek multibagian, lihat Memeriksa integritas objek.

Operasi pengunggahan multibagian serentak

Dalam lingkungan pengembangan terdistribusi, aplikasi Anda dapat memulai beberapa pembaruan pada objek yang sama secara bersamaan. Misalnya, aplikasi Anda mungkin memulai beberapa unggahan multipart dengan menggunakan kunci objek yang sama. Untuk setiap unggahan ini, aplikasi Anda kemudian dapat mengunggah bagian dan mengirim sebuah permintaan menyelesaikan unggahan ke Amazon S3 untuk membuat objek. Untuk S3 Express One Zone, waktu pembuatan objek adalah tanggal penyelesaian unggahan multibagian.

> **Important**
>
> Pembuatan versi tidak didukung untuk objek yang disimpan dalam bucket direktori.

Unggahan dan harga multipart

Setelah Anda memulai unggahan multibagian, Amazon S3 akan menyimpan semua bagian hingga Anda menyelesaikan atau membatalkan unggahan. Sepanjang masa pakainya, Anda akan ditagih untuk semua penyimpanan, bandwidth, dan permintaan untuk unggahan multibagian ini dan bagian terkaitnya. Jika Anda membatalkan unggahan multibagian, Amazon S3 menghapus artefak unggahan dan bagian apa pun yang telah Anda unggah, dan Anda tidak lagi ditagih untuk itu. Tidak ada biaya penghapusan awal untuk menghapus unggahan multibagian yang tidak lengkap, terlepas dari kelas penyimpanan yang ditentukan. Untuk informasi selengkapnya tentang harga, lihat Harga Amazon S3.

> **Important**
>
> Jika permintaan upload multipart lengkap tidak berhasil dikirim, bagian objek tidak dirakit dan objek tidak dibuat. Anda akan ditagih untuk semua penyimpanan yang terkait dengan bagian yang diunggah. Penting bagi Anda untuk menyelesaikan unggahan multibagian agar objek dibuat atau membatalkan unggahan multibagian untuk menghapus bagian yang diunggah. Sebelum dapat menghapus bucket direktori, Anda harus menyelesaikan atau membatalkan semua unggahan multipart yang sedang berlangsung. Bucket direktori tidak mendukung konfigurasi Siklus Hidup S3. Jika perlu, Anda dapat mencantumkan unggahan multibagian aktif, lalu membatalkan unggahan, lalu menghapus bucket.

Operasi dan izin API unggahan multibagian

Untuk mengizinkan akses ke operasi API manajemen objek pada bucket direktori, Anda memberikan `s3express:CreateSession` izin dalam kebijakan bucket atau kebijakan berbasis identitas AWS Identity and Access Management (IAM).

Anda harus memiliki izin yang diperlukan untuk menggunakan operasi pengunggahan multibagian. Anda dapat menggunakan kebijakan bucket atau kebijakan berbasis identitas IAM untuk memberikan izin kepada prinsipal IAM untuk melakukan operasi ini. Tabel berikut mencantumkan izin yang diperlukan untuk berbagai operasi pengunggahan multibagian.

Anda dapat mengidentifikasi inisiator unggahan multipart melalui elemen. `Initiator` Jika inisiator adalah Akun AWS, elemen ini memberikan informasi yang sama dengan `Owner` elemen. Jika inisiator adalah seorang pengguna IAM, elemen ini akan menyediakan ARN pengguna dan nama tampilan.

Tindakan	Izin yang diperlukan
Membuat unggahan multibagian	Untuk membuat unggahan multipart, Anda harus diizinkan untuk melakukan `s3express:CreateSession` tindakan pada bucket direktori.
Memulai unggahan multipart	Untuk memulai unggahan multipart, Anda harus diizinkan untuk melakukan `s3express:CreateSession` tindakan pada bucket direktori.
Unggah bagian	Untuk mengunggah bagian, Anda harus diizinkan untuk melakukan `s3express:CreateSession` tindakan pada bucket direktori. Agar inisiator dapat mengunggah bagian, pemilik bucket harus mengizinkan inisiator untuk melakukan `s3express:CreateSession` tindakan pada bucket direktori.
Unggah bagian (salin)	Untuk mengunggah bagian, Anda harus diizinkan untuk melakukan `s3express:CreateSession` tindakan pada bucket direktori. Agar inisiator dapat mengunggah bagian untuk sebuah objek, pemilik bucket harus mengizinkan inisiator untuk melakukan tindakan `s3express:CreateSession` pada objek.
Selesaikan unggahan multibagian	Untuk menyelesaikan unggahan multibagian, Anda harus diizinkan untuk melakukan `s3express:CreateSession` tindakan pada bucket direktori. Agar inisiator dapat menyelesaikan unggahan multibagian, pemilik bucket harus mengizinkan inisiator untuk melakukan `s3express:CreateSession` tindakan pada objek.
Pembatalan sebuah unggahan multipart	Untuk membatalkan unggahan multibagian, Anda harus diizinkan untuk melakukan tindakan. `s3express:CreateSession` Agar inisiator membatalkan unggahan multibagian, inisiator harus diberikan akses izin eksplisit untuk melakukan tindakan. `s3express:CreateSession`

Tindakan	Izin yang diperlukan
Daftar bagian	Untuk mencantumkan bagian-bagian dalam unggahan multibagian, Anda harus diizinkan untuk melakukan `s3express:CreateSession` tindakan pada bucket direktori.
Membuat daftar unggahan multibagian yang sedang berlangsung	Untuk membuat daftar unggahan multipart yang sedang berlangsung ke bucket, Anda harus diizinkan melakukan `s3:ListBucketMultipartUploads` tindakan pada bucket tersebut.

Dukungan operasi API untuk unggahan multipart

Bagian berikut dalam Referensi API Amazon Simple Storage Service menjelaskan operasi Amazon S3 REST API untuk unggahan multipart.

- CreateMultipartUpload
- UploadPart
- UploadPartCopy
- CompleteMultipartUpload
- AbortMultipartUpload
- ListParts
- ListMultipartUploads

Contoh

Untuk menggunakan unggahan multibagian untuk mengunggah objek ke S3 Express One Zone dalam bucket direktori, lihat contoh berikut.

Topik

- Membuat unggahan multibagian
- Mengunggah bagian-bagian dari unggahan multipart
- Menyelesaikan unggahan multibagian

- Membatalkan unggahan multibagian
- Membuat operasi salin unggahan multibagian
- Mencantumkan unggahan multipart yang sedang berlangsung
- Cantumkan bagian-bagian dari unggahan multipart

Membuat unggahan multibagian

Contoh berikut menunjukkan cara membuat unggahan multipart.

Menggunakan AWS SDK

SDK for Java 2.x

Example

```
/**
 * This method creates a multipart upload request that generates a unique upload ID
 that is used to track
 * all the upload parts
 *
 * @param s3
 * @param bucketName - for example, 'doc-example-bucket--use1-az4--x-s3'
 * @param key
 * @return
 */
 private static String createMultipartUpload(S3Client s3, String bucketName, String
 key) {

     CreateMultipartUploadRequest createMultipartUploadRequest =
 CreateMultipartUploadRequest.builder()
             .bucket(bucketName)
             .key(key)
             .build();

     String uploadId = null;

     try {
         CreateMultipartUploadResponse response =
 s3.createMultipartUpload(createMultipartUploadRequest);
         uploadId = response.uploadId();
     }
     catch (S3Exception e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return uploadId;
```

SDK for Python

Example

```
def create_multipart_upload(s3_client, bucket_name, key_name):
    '''
   Create a multipart upload to a directory bucket

   :param s3_client: boto3 S3 client
   :param bucket_name: The destination bucket for the multipart upload
   :param key_name: The key name for the object to be uploaded
   :return: The UploadId for the multipart upload if created successfully, else None
   '''

   try:
        mpu = s3_client.create_multipart_upload(Bucket = bucket_name, Key =
 key_name)
        return mpu['UploadId']
    except ClientError as e:
        logging.error(e)
        return None
```

Menggunakan AWS CLI

Contoh ini menunjukkan cara membuat unggahan multipart ke bucket direktori dengan menggunakan file. AWS CLI*Perintah ini memulai unggahan multipart ke direktori bucket bucket-base-name-- azid --x-s3 untuk objek KEY_NAME.* Untuk menggunakan perintah, ganti *placeholder input pengguna dengan informasi* Anda sendiri.

```
aws s3api create-multipart-upload --bucket bucket-base-name--azid--x-s3 --key KEY_NAME
```

Untuk informasi lebih lanjut, lihat create-multipart-uploaddi AWS Command Line Interface.

Mengunggah bagian-bagian dari unggahan multipart

Contoh berikut menunjukkan cara mengunggah bagian dari unggahan multipart.

Menggunakan AWS SDK

SDK for Java 2.x

Contoh berikut menunjukkan cara memecah satu objek menjadi beberapa bagian dan kemudian mengunggah bagian-bagian tersebut ke bucket direktori dengan menggunakan SDK for Java 2.x.

Example

```
/**
 * This method creates part requests and uploads individual parts to S3 and then
 returns all the completed parts
 *
 * @param s3
 * @param bucketName
 * @param key
 * @param uploadId
 * @throws IOException
 */
 private static ListCompletedPartmultipartUpload(S3Client s3, String bucketName,
 String key, String uploadId, String filePath) throws IOException {

        int partNumber = 1;
        ListCompletedPart completedParts = new ArrayList<>();
        ByteBuffer bb = ByteBuffer.allocate(1024 * 1024 * 5); // 5 MB byte buffer

        // read the local file, breakdown into chunks and process
        try (RandomAccessFile file = new RandomAccessFile(filePath, "r")) {
            long fileSize = file.length();
            int position = 0;
            while (position < fileSize) {
                file.seek(position);
                int read = file.getChannel().read(bb);

                bb.flip(); // Swap position and limit before reading from the
 buffer.
                UploadPartRequest uploadPartRequest = UploadPartRequest.builder()
                        .bucket(bucketName)
                        .key(key)
                        .uploadId(uploadId)
                        .partNumber(partNumber)
                        .build();

                UploadPartResponse partResponse = s3.uploadPart(
```

```
                        uploadPartRequest,
                        RequestBody.fromByteBuffer(bb));

                CompletedPart part = CompletedPart.builder()
                        .partNumber(partNumber)
                        .eTag(partResponse.eTag())
                        .build();
                completedParts.add(part);

                bb.clear();
                position += read;
                partNumber++;
            }
        }

        catch (IOException e) {
            throw e;
        }
        return completedParts;
    }
```

SDK for Python

Contoh berikut menunjukkan cara memecah satu objek menjadi beberapa bagian dan kemudian mengunggah bagian-bagian tersebut ke bucket direktori dengan menggunakan SDK untuk Python.

Example

```
def multipart_upload(s3_client, bucket_name, key_name, mpu_id, part_size):
    '''
    Break up a file into multiple parts and upload those parts to a directory bucket

    :param s3_client: boto3 S3 client
    :param bucket_name: Destination bucket for the multipart upload
    :param key_name: Key name for object to be uploaded and for the local file
 that's being uploaded
    :param mpu_id: The UploadId returned from the create_multipart_upload call
    :param part_size: The size parts that the object will be broken into, in bytes.
                      Minimum 5 MiB, Maximum 5 GiB. There is no minimum size for the
 last part of your multipart upload.
    :return: part_list for the multipart upload if all parts are uploaded
 successfully, else None
```

```
    '''

    part_list = []
    try:
        with open(key_name, 'rb') as file:
            part_counter = 1
            while True:
                file_part = file.read(part_size)
                if not len(file_part):
                    break
                upload_part = s3_client.upload_part(
                    Bucket = bucket_name,
                    Key = key_name,
                    UploadId = mpu_id,
                    Body = file_part,
                    PartNumber = part_counter
                )
                part_list.append({'PartNumber': part_counter, 'ETag':
 upload_part['ETag']})
                part_counter += 1
    except ClientError as e:
        logging.error(e)
        return None
    return part_list
```

Menggunakan AWS CLI

Contoh ini menunjukkan cara memecah satu objek menjadi beberapa bagian dan kemudian mengunggah bagian-bagian tersebut ke bucket direktori dengan menggunakan file AWS CLI. Untuk menggunakan perintah, ganti *placeholder input pengguna dengan informasi* Anda sendiri.

```
aws s3api upload-part --bucket bucket-base-name--azid--x-s3 --
key KEY_NAME --part-number 1 --body LOCAL_FILE_NAME --upload-id
 "AS_mgt9RaQE9GEaifATue15dAAAAAAAAAAEMAAAAAAAAADQwNzI4MDU0MjUyMBYAAAAAAAAAAA0AAAAAAAAAAAH2AfYAA
```

Untuk informasi selengkapnya, lihat bagian unggahan di. AWS Command Line Interface

Menyelesaikan unggahan multibagian

Contoh berikut menunjukkan cara menyelesaikan unggahan multipart.

Menggunakan AWS SDK

SDK for Java 2.x

Contoh berikut menunjukkan cara menyelesaikan upload multipart dengan menggunakan SDK for Java 2.x.

Example

```
/**
 * This method completes the multipart upload request by collating all the upload
 parts
 * @param s3
 * @param bucketName - for example, 'doc-example-bucket--usw2-az1--x-s3'
 * @param key
 * @param uploadId
 * @param uploadParts
 */
 private static void completeMultipartUpload(S3Client s3, String bucketName, String
 key, String uploadId, ListCompletedPart uploadParts) {
        CompletedMultipartUpload completedMultipartUpload =
 CompletedMultipartUpload.builder()
                .parts(uploadParts)
                .build();

        CompleteMultipartUploadRequest completeMultipartUploadRequest =
                CompleteMultipartUploadRequest.builder()
                        .bucket(bucketName)
                        .key(key)
                        .uploadId(uploadId)
                        .multipartUpload(completedMultipartUpload)
                        .build();

        s3.completeMultipartUpload(completeMultipartUploadRequest);
    }

    public static void multipartUploadTest(S3Client s3, String bucketName, String
 key, String localFilePath)  {
        System.out.println("Starting multipart upload for: " + key);
        try {
            String uploadId = createMultipartUpload(s3, bucketName, key);
            System.out.println(uploadId);
            ListCompletedPart parts = multipartUpload(s3, bucketName, key, uploadId,
 localFilePath);
```

```
            completeMultipartUpload(s3, bucketName, key, uploadId, parts);
            System.out.println("Multipart upload completed for: " + key);
        }

        catch (Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

SDK for Python

Contoh berikut menunjukkan cara menyelesaikan upload multipart dengan menggunakan SDK untuk Python.

Example

```
def complete_multipart_upload(s3_client, bucket_name, key_name, mpu_id, part_list):
    '''
    Completes a multipart upload to a directory bucket

    :param s3_client: boto3 S3 client
    :param bucket_name: The destination bucket for the multipart upload
    :param key_name: The key name for the object to be uploaded
    :param mpu_id: The UploadId returned from the create_multipart_upload call
    :param part_list: The list of uploaded part numbers with their associated ETags
    :return: True if the multipart upload was completed successfully, else False
    '''

    try:
        s3_client.complete_multipart_upload(
            Bucket = bucket_name,
            Key = key_name,
            UploadId = mpu_id,
            MultipartUpload = {
                'Parts': part_list
            }
        )
    except ClientError as e:
        logging.error(e)
        return False
    return True

if __name__ == '__main__':
```

```
    MB = 1024 ** 2
    region = 'us-west-2'
    bucket_name = 'BUCKET_NAME'
    key_name = 'OBJECT_NAME'
    part_size = 10 * MB
    s3_client = boto3.client('s3', region_name = region)
    mpu_id = create_multipart_upload(s3_client, bucket_name, key_name)
    if mpu_id is not None:
        part_list = multipart_upload(s3_client, bucket_name, key_name, mpu_id,
 part_size)
        if part_list is not None:
            if complete_multipart_upload(s3_client, bucket_name, key_name, mpu_id,
 part_list):
                print (f'{key_name} successfully uploaded through a ultipart upload
 to {bucket_name}')
            else:
                print (f'Could not upload {key_name} hrough a multipart upload to
 {bucket_name}')
```

Menggunakan AWS CLI

Contoh ini menunjukkan cara menyelesaikan unggahan multipart untuk bucket direktori dengan menggunakan file. AWS CLI Untuk menggunakan perintah, ganti *placeholder input pengguna dengan informasi* Anda sendiri.

```
aws s3api complete-multipart-upload --bucket bucket-base-name--azid--x-s3 --
key KEY_NAME --upload-id
 "AS_mgt9RaQE9GEaifATue15dAAAAAAAAAAEMAAAAAAAAADQwNzI4MDU0MjUyMBYAAAAAAAAAAA0AAAAAAAAAAAH2AfYAA
 --multipart-upload file://parts.json
```

Contoh ini mengambil struktur JSON yang menjelaskan bagian-bagian dari upload multipart yang harus dipasang kembali ke dalam file lengkap. Dalam contoh ini, `file://` awalan digunakan untuk memuat struktur JSON dari file di folder lokal bernama. `parts`

parts.json:

```
parts.json
{
  "Parts": [
    {
      "ETag": "6b78c4a64dd641a58dac8d9258b88147",
```

```
      "PartNumber": 1
    }
  ]
}
```

Untuk informasi lebih lanjut, lihat complete-multipart-uploaddi AWS Command Line Interface.

Membatalkan unggahan multibagian

Contoh berikut menunjukkan cara membatalkan unggahan multipart.

Menggunakan AWS SDK

SDK for Java 2.x

Contoh berikut menunjukkan cara membatalkan upload multipart dengan menggunakan SDK for Java 2.x.

Example

```
public static void abortMultiPartUploads( S3Client s3, String bucketName ) {

         try {
             ListMultipartUploadsRequest listMultipartUploadsRequest =
 ListMultipartUploadsRequest.builder()
                     .bucket(bucketName)
                     .build();

             ListMultipartUploadsResponse response =
 s3.listMultipartUploads(listMultipartUploadsRequest);
             ListMultipartUpload uploads = response.uploads();

             AbortMultipartUploadRequest abortMultipartUploadRequest;
             for (MultipartUpload upload: uploads) {
                 abortMultipartUploadRequest = AbortMultipartUploadRequest.builder()
                         .bucket(bucketName)
                         .key(upload.key())
                         .uploadId(upload.uploadId())
                         .build();

                 s3.abortMultipartUpload(abortMultipartUploadRequest);
             }

         }
```

```
        catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

SDK for Python

Contoh berikut menunjukkan cara membatalkan upload multipart dengan menggunakan SDK untuk Python.

Example

```
import logging
import boto3
from botocore.exceptions import ClientError


def abort_multipart_upload(s3_client, bucket_name, key_name, upload_id):
    '''
    Aborts a partial multipart upload in a directory bucket.

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket where the multipart upload was initiated - for
 example, 'doc-example-bucket--usw2-az1--x-s3'
    :param key_name: Name of the object for which the multipart upload needs to be
 aborted
    :param upload_id: Multipart upload ID for the multipart upload to be aborted
    :return: True if the multipart upload was successfully aborted, False if not
    '''
    try:
        s3_client.abort_multipart_upload(
            Bucket = bucket_name,
            Key = key_name,
            UploadId = upload_id
        )
    except ClientError as e:
        logging.error(e)
        return False
    return True


if __name__ == '__main__':
```

```
    region = 'us-west-2'
    bucket_name = 'BUCKET_NAME'
    key_name = 'KEY_NAME'
        upload_id = 'UPLOAD_ID'
    s3_client = boto3.client('s3', region_name = region)
    if abort_multipart_upload(s3_client, bucket_name, key_name, upload_id):
        print (f'Multipart upload for object {key_name} in {bucket_name} bucket has
been aborted')
    else:
        print (f'Unable to abort multipart upload for object {key_name} in
{bucket_name} bucket')
```

Menggunakan AWS CLI

Contoh berikut menunjukkan cara membatalkan upload multipart dengan menggunakan file. AWS CLI Untuk menggunakan perintah, ganti *placeholder input pengguna dengan informasi* Anda sendiri.

```
aws s3api abort-multipart-upload --bucket bucket-base-name--azid--x-s3 --key KEY_NAME
 --upload-id
 "AS_mgt9RaQE9GEaifATue15dAAAAAAAAAAEMAAAAAAAAADQwNzI4MDU0MjUyMBYAAAAAAAAAAA0AAAAAAAAAAAH2AfYAA
MAQAAAAB0OxUFeA7LTbWWFS8WYwhrxDxTIDN-pdEEq_agIHqsbg"
```

Untuk informasi lebih lanjut, lihat abort-multipart-uploaddi AWS Command Line Interface.

Membuat operasi salin unggahan multibagian

Contoh berikut menunjukkan cara menyalin bjects dari satu bucket ke bucket lainnya menggunakan unggahan multipart.

Menggunakan AWS SDK

SDK for Java 2.x

Contoh berikut menunjukkan cara menggunakan unggahan multipart untuk menyalin objek secara terprogram dari satu bucket ke bucket lainnya dengan menggunakan SDK for Java 2.x.

Example

```
/**
 * This method creates a multipart upload request that generates a unique upload ID
 that is used to track
```

```
 * all the upload parts.
 *
 * @param s3
 * @param bucketName
 * @param key
 * @return
 */
private static String createMultipartUpload(S3Client s3, String bucketName, String
key) {
       CreateMultipartUploadRequest createMultipartUploadRequest =
CreateMultipartUploadRequest.builder()
               .bucket(bucketName)
               .key(key)
               .build();
       String uploadId = null;
       try {
           CreateMultipartUploadResponse response =
s3.createMultipartUpload(createMultipartUploadRequest);
           uploadId = response.uploadId();
       } catch (S3Exception e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
       return uploadId;
 }

 /**
  * Creates copy parts based on source object size and copies over individual parts
  *
  * @param s3
  * @param sourceBucket
  * @param sourceKey
  * @param destnBucket
  * @param destnKey
  * @param uploadId
  * @return
  * @throws IOException
  */
   public static ListCompletedPart multipartUploadCopy(S3Client s3, String
sourceBucket, String sourceKey, String destnBucket, String destnKey, String
uploadId) throws IOException {

       // Get the object size to track the end of the copy operation.
       HeadObjectRequest headObjectRequest = HeadObjectRequest
```

```
                .builder()
                .bucket(sourceBucket)
                .key(sourceKey)
                .build();
        HeadObjectResponse response = s3.headObject(headObjectRequest);
        Long objectSize = response.contentLength();

        System.out.println("Source Object size: " + objectSize);

        // Copy the object using 20 MB parts.
        long partSize = 20 * 1024 * 1024;
        long bytePosition = 0;
        int partNum = 1;
        ListCompletedPart completedParts = new ArrayList<>();
        while (bytePosition < objectSize) {
            // The last part might be smaller than partSize, so check to make sure
            // that lastByte isn't beyond the end of the object.
            long lastByte = Math.min(bytePosition + partSize - 1, objectSize - 1);

            System.out.println("part no: " + partNum + ", bytePosition: " +
bytePosition + ", lastByte: " + lastByte);

            // Copy this part.
            UploadPartCopyRequest req = UploadPartCopyRequest.builder()
                    .uploadId(uploadId)
                    .sourceBucket(sourceBucket)
                    .sourceKey(sourceKey)
                    .destinationBucket(destnBucket)
                    .destinationKey(destnKey)
                    .copySourceRange("bytes="+bytePosition+"-"+lastByte)
                    .partNumber(partNum)
                    .build();
            UploadPartCopyResponse res = s3.uploadPartCopy(req);
            CompletedPart part = CompletedPart.builder()
                    .partNumber(partNum)
                    .eTag(res.copyPartResult().eTag())
                    .build();
            completedParts.add(part);
            partNum++;
            bytePosition += partSize;
        }
        return completedParts;
    }
```

```
    public static void multipartCopyUploadTest(S3Client s3, String srcBucket, String
 srcKey, String destnBucket, String destnKey)  {
        System.out.println("Starting multipart copy for: " + srcKey);
        try {
            String uploadId = createMultipartUpload(s3, destnBucket, destnKey);
            System.out.println(uploadId);
            ListCompletedPart parts = multipartUploadCopy(s3, srcBucket,
 srcKey,destnBucket,  destnKey, uploadId);
            completeMultipartUpload(s3, destnBucket, destnKey, uploadId, parts);
            System.out.println("Multipart copy completed for: " + srcKey);
        } catch (Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

SDK for Python

Contoh berikut menunjukkan cara menggunakan unggahan multipart untuk menyalin objek secara terprogram dari satu bucket ke bucket lainnya dengan menggunakan SDK untuk Python.

Example

```
import logging
import boto3
from botocore.exceptions import ClientError

def head_object(s3_client, bucket_name, key_name):
    '''
    Returns metadata for an object in a directory bucket

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket that contains the object to query for metadata
    :param key_name: Key name to query for metadata
    :return: Metadata for the specified object if successful, else None
    '''

    try:
        response = s3_client.head_object(
            Bucket = bucket_name,
            Key = key_name
        )
        return response
```

```
    except ClientError as e:
        logging.error(e)
        return None

def create_multipart_upload(s3_client, bucket_name, key_name):
    '''
    Create a multipart upload to a directory bucket

    :param s3_client: boto3 S3 client
    :param bucket_name: Destination bucket for the multipart upload
    :param key_name: Key name of the object to be uploaded
    :return: UploadId for the multipart upload if created successfully, else None
    '''

    try:
        mpu = s3_client.create_multipart_upload(Bucket = bucket_name, Key =
 key_name)
        return mpu['UploadId']
    except ClientError as e:
        logging.error(e)
        return None

def multipart_copy_upload(s3_client, source_bucket_name, key_name,
 target_bucket_name, mpu_id, part_size):
    '''
    Copy an object in a directory bucket to another bucket in multiple parts of a
 specified size

    :param s3_client: boto3 S3 client
    :param source_bucket_name: Bucket where the source object exists
    :param key_name: Key name of the object to be copied
    :param target_bucket_name: Destination bucket for copied object
    :param mpu_id: The UploadId returned from the create_multipart_upload call
    :param part_size: The size parts that the object will be broken into, in bytes.
                      Minimum 5 MiB, Maximum 5 GiB. There is no minimum size for the
 last part of your multipart upload.
    :return: part_list for the multipart copy if all parts are copied successfully,
 else None
    '''

    part_list = []
    copy_source = {
        'Bucket': source_bucket_name,
        'Key': key_name
```

```
    }
    try:
        part_counter = 1
        object_size = head_object(s3_client, source_bucket_name, key_name)
        if object_size is not None:
            object_size = object_size['ContentLength']
        while (part_counter - 1) * part_size <object_size:
            bytes_start = (part_counter - 1) * part_size
            bytes_end = (part_counter * part_size) - 1
            upload_copy_part = s3_client.upload_part_copy (
                Bucket = target_bucket_name,
                CopySource = copy_source,
                CopySourceRange = f'bytes={bytes_start}-{bytes_end}',
                Key = key_name,
                PartNumber = part_counter,
                UploadId = mpu_id
            )
            part_list.append({'PartNumber': part_counter, 'ETag':
 upload_copy_part['CopyPartResult']['ETag']})
            part_counter += 1
    except ClientError as e:
        logging.error(e)
        return None
    return part_list

def complete_multipart_upload(s3_client, bucket_name, key_name, mpu_id, part_list):
    '''
    Completes a multipart upload to a directory bucket

    :param s3_client: boto3 S3 client
    :param bucket_name: Destination bucket for the multipart upload
    :param key_name: Key name of the object to be uploaded
    :param mpu_id: The UploadId returned from the create_multipart_upload call
    :param part_list: List of uploaded part numbers with associated ETags
    :return: True if the multipart upload was completed successfully, else False
    '''

    try:
        s3_client.complete_multipart_upload(
            Bucket = bucket_name,
            Key = key_name,
            UploadId = mpu_id,
            MultipartUpload = {
                'Parts': part_list
```

```
            }
        )
    except ClientError as e:
        logging.error(e)
        return False
    return True

if __name__ == '__main__':
    MB = 1024 ** 2
    region = 'us-west-2'
    source_bucket_name = 'SOURCE_BUCKET_NAME'
    target_bucket_name = 'TARGET_BUCKET_NAME'
    key_name = 'KEY_NAME'
    part_size = 10 * MB
    s3_client = boto3.client('s3', region_name = region)
    mpu_id = create_multipart_upload(s3_client, target_bucket_name, key_name)
    if mpu_id is not None:
        part_list = multipart_copy_upload(s3_client, source_bucket_name, key_name,
 target_bucket_name, mpu_id, part_size)
        if part_list is not None:
            if complete_multipart_upload(s3_client, target_bucket_name, key_name,
 mpu_id, part_list):
                print (f'{key_name} successfully copied through multipart copy from
 {source_bucket_name} to {target_bucket_name}')
            else:
                print (f'Could not copy {key_name} through multipart copy from
 {source_bucket_name} to {target_bucket_name}')
```

Menggunakan AWS CLI

Contoh berikut menunjukkan cara menggunakan unggahan multibagian untuk menyalin objek secara terprogram dari satu bucket ke bucket direktori menggunakan file. AWS CLI Untuk menggunakan perintah, ganti *placeholder input pengguna dengan informasi* Anda sendiri.

```
aws s3api upload-part-copy --bucket bucket-base-name--azid--x-s3 --key TARGET_KEY_NAME
 --copy-source SOURCE_BUCKET_NAME/SOURCE_KEY_NAME --part-number 1 --upload-id
 "AS_mgt9RaQE9GEaifATue15dAAAAAAAAAAEMAAAAAAAAADQwNzI4MDU0MjUyMBYAAAAAAAAAAA0AAAAAAAAAAAH2AfYAA
```

Untuk informasi lebih lanjut, lihat upload-part-copydi AWS Command Line Interface.

Mencantumkan unggahan multipart yang sedang berlangsung

Untuk mencantumkan unggahan multibagian yang sedang berlangsung ke bucket direktori, Anda dapat menggunakan AWS SDK, atau file. AWS CLI

Menggunakan AWS SDK

SDK for Java 2.x

Contoh berikut menunjukkan cara membuat daftar unggahan multipart yang sedang berlangsung (tidak lengkap) dengan menggunakan SDK for Java 2.x.

Example

```
public static void listMultiPartUploads( S3Client s3, String bucketName) {
       try {
           ListMultipartUploadsRequest listMultipartUploadsRequest =
ListMultipartUploadsRequest.builder()
               .bucket(bucketName)
               .build();

           ListMultipartUploadsResponse response =
s3.listMultipartUploads(listMultipartUploadsRequest);
           List MultipartUpload uploads = response.uploads();
           for (MultipartUpload upload: uploads) {
               System.out.println("Upload in progress: Key = \"" + upload.key() +
"\", id = " + upload.uploadId());
           }
     }
     catch (S3Exception e) {
           System.err.println(e.getMessage());
           System.exit(1);
     }
 }
```

SDK for Python

Contoh berikut menunjukkan cara membuat daftar unggahan multipart yang sedang berlangsung (tidak lengkap) dengan menggunakan SDK untuk Python.

Example

```
import logging
```

```
import boto3
from botocore.exceptions import ClientError

def list_multipart_uploads(s3_client, bucket_name):
    '''
    List any incomplete multipart uploads in a directory bucket in e specified gion

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket to check for incomplete multipart uploads
    :return: List of incomplete multipart uploads if there are any, None if not
    '''

    try:
        response = s3_client.list_multipart_uploads(Bucket = bucket_name)
        if 'Uploads' in response.keys():
            return response['Uploads']
        else:
            return None
    except ClientError as e:
        logging.error(e)

if __name__ == '__main__':
    bucket_name = 'BUCKET_NAME'
    region = 'us-west-2'
    s3_client = boto3.client('s3', region_name = region)
    multipart_uploads = list_multipart_uploads(s3_client, bucket_name)
    if multipart_uploads is not None:
        print (f'There are {len(multipart_uploads)} ncomplete multipart uploads for
 {bucket_name}')
    else:
        print (f'There are no incomplete multipart uploads for {bucket_name}')
```

Menggunakan AWS CLI

Contoh berikut menunjukkan cara membuat daftar unggahan multipart yang sedang berlangsung (tidak lengkap) dengan menggunakan. AWS CLI Untuk menggunakan perintah ganti *placeholder input pengguna dengan informasi* Anda sendiri.

```
aws s3api list-multipart-uploads --bucket bucket-base-name--azid--x-s3
```

Untuk informasi lebih lanjut, lihat list-multipart-uploadsdi AWS Command Line Interface.

Cantumkan bagian-bagian dari unggahan multipart

Contoh berikut menunjukkan cara membuat daftar bagian dari unggahan multipart ke bucket direktori.

Menggunakan AWS SDK

SDK for Java 2.x

Contoh berikut menunjukkan cara membuat daftar bagian dari unggahan multipart ke bucket direktori dengan menggunakan SDK for Java 2.x.

```
public static void listMultiPartUploadsParts( S3Client s3, String bucketName, String
 objKey, String uploadID) {

        try {
            ListPartsRequest listPartsRequest = ListPartsRequest.builder()
                .bucket(bucketName)
                .uploadId(uploadID)
                .key(objKey)
                .build();

            ListPartsResponse response = s3.listParts(listPartsRequest);
            ListPart parts = response.parts();
            for (Part part: parts) {
                System.out.println("Upload in progress: Part number = \"" +
 part.partNumber() + "\", etag = " + part.eTag());
            }

        }

        catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }


    }
```

SDK for Python

Contoh berikut menunjukkan cara membuat daftar bagian dari unggahan multipart ke bucket direktori dengan menggunakan SDK untuk Python.

```
import logging
import boto3
from botocore.exceptions import ClientError

def list_parts(s3_client, bucket_name, key_name, upload_id):
    '''
    Lists the parts that have been uploaded for a specific multipart upload to a
 directory bucket.

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket that multipart uploads parts have been uploaded to
    :param key_name: Name of the object that has parts uploaded
    :param upload_id: Multipart upload ID that the parts are associated with
    :return: List of parts associated with the specified multipart upload, None if
 there are no parts
    '''
    parts_list = []
    next_part_marker = ''
    continuation_flag = True
    try:
        while continuation_flag:
            if next_part_marker == '':
                response = s3_client.list_parts(
                    Bucket = bucket_name,
                    Key = key_name,
                    UploadId = upload_id
                )
            else:
                response = s3_client.list_parts(
                    Bucket = bucket_name,
                    Key = key_name,
                    UploadId = upload_id,
                    NextPartMarker = next_part_marker
                )
            if 'Parts' in response:
                for part in response['Parts']:
                    parts_list.append(part)
                if response['IsTruncated']:
                    next_part_marker = response['NextPartNumberMarker']
                else:
                    continuation_flag = False
            else:
                continuation_flag = False
```

```
        return parts_list
    except ClientError as e:
        logging.error(e)
        return None

if __name__ == '__main__':
    region = 'us-west-2'
    bucket_name = 'BUCKET_NAME'
    key_name = 'KEY_NAME'
    upload_id = 'UPLOAD_ID'
    s3_client = boto3.client('s3', region_name = region)
    parts_list = list_parts(s3_client, bucket_name, key_name, upload_id)
    if parts_list is not None:
        print (f'{key_name} has {len(parts_list)} parts uploaded to {bucket_name}')
    else:
        print (f'There are no multipart uploads with that upload ID for
 {bucket_name} bucket')
```

Menggunakan AWS CLI

Contoh berikut menunjukkan cara membuat daftar bagian dari unggahan multipart ke bucket direktori dengan menggunakan. AWS CLI Untuk menggunakan perintah ganti *placeholder input pengguna dengan informasi* Anda sendiri.

```
aws s3api list-parts --bucket bucket-base-name--azid--x-s3 --key KEY_NAME --upload-id
 "AS_mgt9RaQE9GEaifATue15dAAAAAAAAAAEMAAAAAAAAADQwNzI4MDU0MjUyMBYAAAAAAAAAAA0AAAAAAAAAAAH2AfYAA
```

Untuk informasi selengkapnya, lihat bagian daftar di. AWS Command Line Interface

Menyalin objek ke bucket direktori

Operasi penyalinan membuat salinan objek yang sudah disimpan di Amazon S3. Anda dapat menyalin objek antara bucket direktori dan bucket tujuan umum. Anda juga dapat menyalin objek di dalam bucket dan di bucket dengan tipe yang sama, misalnya, dari bucket direktori ke bucket direktori.

Anda dapat membuat salinan objek hingga 5 GB dalam satu operasi atom. Namun, untuk menyalin objek yang lebih besar dari 5 GB, Anda harus menggunakan operasi API unggahan multibagian. Untuk informasi selengkapnya, lihat Menggunakan unggahan multibagian dengan bucket direktori.

Izin

Untuk menyalin objek, Anda harus memiliki izin berikut:

- Untuk menyalin objek dari satu bucket direktori ke bucket direktori lain, Anda harus memiliki izin `s3express:CreateSession`.
- Untuk menyalin objek dari bucket direktori ke bucket tujuan umum, Anda harus memiliki izin `s3express:CreateSession` dan izin `s3:PutObject`untuk menulis salinan objek ke bucket tujuan.
- Untuk menyalin objek dari bucket tujuan umum ke bucket direktori, Anda harus memiliki `s3express:CreateSession` izin dan `s3:GetObject` izin untuk membaca objek sumber yang sedang disalin.

  Untuk informasi selengkapnya, lihat CopyObject dalam Referensi API Amazon Simple Storage Service.

Enkripsi

Amazon S3 secara otomatis mengenkripsi semua objek baru yang diunggah ke bucket S3. Konfigurasi enkripsi default bucket S3 selalu diaktifkan dan diatur secara minimum ke enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3).

Untuk bucket direktori, hanya SSE-S3 yang didukung. Untuk bucket tujuan umum, Anda dapat menggunakan SSE-S3 (default), enkripsi sisi server dengan () kunci (SSE-KMS), enkripsi sisi server dua lapis dengan AWS Key Management Service kunci (DSSE-KMS), atau enkripsi sisi server dengan AWS KMS kunci yang disediakan pelanggan (SSE-C).AWS KMS

Jika Anda membuat permintaan salinan yang menetapkan parameter SSE-C, SSE-KMS, atau DSSE-KMS pada bucket direktori sebagai sumber atau tujuan, respons akan mengembalikan kesalahan,

Tanda

Bucket direktori tidak mendukung tanda. Jika Anda menyalin objek yang memiliki tag dari bucket tujuan umum ke bucket direktori, Anda akan menerima `501 (Not Implemented)` respons HTTP. Untuk informasi selengkapnya, lihat CopyObject dalam Referensi API Amazon Simple Storage Service.

ETag

Tag entitas (ETag) untuk S3 Express One Zone adalah string alfanumerik acak dan bukan checksum MD5. Untuk membantu memastikan integritas objek, gunakan checksum tambahan.

Checksum tambahan

S3 Express One Zone menawarkan kepada Anda opsi untuk memilih algoritma checksum yang digunakan untuk memvalidasi data Anda selama mengunggah atau mengunduh. Anda dapat memilih salah satu algoritma pemeriksaan integritas data Secure Hash Algorithms (SHA) atau Cyclic Redundancy Check (CRC) berikut: CRC32, CRC32C, SHA-1, dan SHA-256. Checksum berbasis MD5 tidak didukung dengan kelas penyimpanan S3 Express One Zone.

Untuk informasi selengkapnya, lihat Praktik terbaik checksum tambahan S3.

Fitur yang didukung

Untuk informasi selengkapnya tentang fitur Amazon S3 mana yang didukung untuk S3 Express One Zone, lihat. Apa yang membuat S3 Express One Zone berbeda?

Menggunakan konsol S3 (salin ke bucket direktori)

Untuk menyalin objek dari bucket tujuan umum atau bucket direktori ke bucket direktori

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih bucket tempat Anda ingin menyalin objek dari:
   - Untuk menyalin dari ember tujuan umum, pilih tab Bucket tujuan umum.
   - Untuk menyalin dari bucket direktori, pilih tab Directory buckets.
4. Pilih bucket tujuan umum atau bucket direktori yang berisi objek yang ingin Anda salin.
5. Pilih tab Objek. Pada halaman Objek, pilih kotak centang di sebelah kiri nama objek yang ingin Anda salin.
6. Pada menu Tindakan, pilih Salin.

   Halaman Salin muncul.
7. Di bawah Tujuan, pilih Ember direktori untuk jenis tujuan Anda. Untuk menentukan jalur tujuan, pilih Browse S3, navigasikan ke tujuan, lalu pilih tombol opsi di sebelah kiri tujuan. Pilih Pilih tujuan di sudut kanan bawah.

   Atau, masukkan jalur tujuan.
8. Di bawah Checksum, pilih apakah Anda ingin menyalin objek dengan fungsi checksum yang ada atau ganti fungsi checksum yang ada dengan yang baru. Saat Anda mengunggah objek,

Anda memiliki opsi untuk menentukan algoritma checksum yang digunakan untuk memverifikasi integritas data. Saat menyalin objek, Anda memiliki opsi untuk memilih fungsi baru. Jika Anda awalnya tidak menentukan checksum tambahan, Anda dapat menggunakan bagian e Checksum untuk menambahkannya.

> **Note**
>
> Bahkan jika Anda memilih untuk menggunakan fungsi checksum yang sama, nilai checksum Anda mungkin berubah jika objek berukuran lebih dari 16 MB. Nilai checksum mungkin berubah karena cara checksum dihitung untuk unggahan multibagian. Untuk informasi selengkapnya tentang perubahan checksum saat menyalin objek, lihat Menggunakan checksum tingkat bagian untuk unggahan multibagian.

Untuk mengubah fungsi checksum, pilih Ganti dengan fungsi checksum baru. Pilih fungsi checksum baru dari daftar dropdown. Ketika objek disalin, checksum baru dihitung dan disimpan dengan menggunakan algoritma yang ditentukan.

9. Pilih Salin di sudut kanan bawah. Amazon S3 menyalin objek Anda ke tujuan.

Menggunakan konsol S3 (salin ke bucket tujuan umum)

Untuk menyalin objek dari bucket direktori ke bucket tujuan umum

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih tab Ember direktori.
4. Pilih bucket direktori yang berisi objek yang ingin Anda salin.
5. Pilih tab Objek. Pada halaman Objek, pilih kotak centang di sebelah kiri nama objek yang ingin Anda salin.
6. Pada menu Tindakan, pilih Salin.
7. Di bawah Tujuan, pilih Bucket tujuan umum untuk jenis tujuan Anda. Untuk menentukan jalur tujuan, pilih Browse S3, navigasikan ke tujuan, dan pilih tombol opsi di sebelah kiri tujuan. Pilih Pilih tujuan di sudut kanan bawah.

Atau, masukkan jalur tujuan.

8. Di bawah Checksum, pilih apakah Anda ingin menyalin objek dengan fungsi checksum yang ada atau ganti fungsi checksum yang ada dengan yang baru. Saat Anda mengunggah objek, Anda memiliki opsi untuk menentukan algoritma checksum yang digunakan untuk memverifikasi integritas data. Saat menyalin objek, Anda memiliki opsi untuk memilih fungsi baru. Jika awalnya Anda tidak menentukan checksum tambahan, Anda dapat menggunakan bagian Checksum untuk menambahkannya.

> **Note**
>
> Bahkan jika Anda memilih untuk menggunakan fungsi checksum yang sama, nilai checksum Anda mungkin berubah jika objek berukuran lebih dari 16 MB. Nilai checksum mungkin berubah karena cara checksum dihitung untuk unggahan multibagian. Untuk informasi selengkapnya tentang perubahan checksum saat menyalin objek, lihat Menggunakan checksum tingkat bagian untuk unggahan multibagian.

Untuk mengubah fungsi checksum, pilih Ganti dengan fungsi checksum baru. Pilih fungsi checksum baru dari daftar dropdown. Ketika objek disalin, checksum baru dihitung dan disimpan dengan menggunakan algoritma yang ditentukan.

9. Pilih Salin di sudut kanan bawah. Amazon S3 menyalin objek Anda ke tujuan.

Menggunakan AWS SDK

SDK for Java 2.x

Example

```
public static void copyBucketObject (S3Client s3, String sourceBucket, String
objectKey, String targetBucket) {
     CopyObjectRequest copyReq = CopyObjectRequest.builder()
         .sourceBucket(sourceBucket)
         .sourceKey(objectKey)
         .destinationBucket(targetBucket)
         .destinationKey(objectKey)
         .build();
      String temp = "";
```

```
        try {
            CopyObjectResponse copyRes = s3.copyObject(copyReq);
            System.out.println("Successfully copied " + objectKey +" from bucket " +
sourceBucket +" into bucket "+targetBucket);
        }

        catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
}
```

Menggunakan AWS CLI

`copy-object`Contoh perintah berikut menunjukkan bagaimana Anda dapat menggunakan AWS CLI untuk menyalin objek dari satu bucket ke bucket lain. Anda dapat menyalin objek di antara jenis bucket. Untuk menjalankan perintah ini, ganti placeholder input pengguna dengan informasi Anda sendiri.

```
aws s3api copy-object --copy-source bucket SOURCE_BUCKET/SOURCE_KEY_NAME --
key TARGET_KEY_NAME --bucket TARGET_BUCKET_NAME
```

Untuk informasi selengkapnya, lihat copy-object dalam AWS CLI Referensi Perintah.

Menghapus objek dalam bucket direktori

Anda dapat menghapus objek dari bucket direktori Amazon S3 dengan menggunakan konsol Amazon S3 AWS Command Line Interface ,AWS CLI(), atau SDK. AWS Untuk informasi selengkapnya, lihat Ember direktori dan Apa itu S3 Express One Zone?.

> **Warning**
>
> - Menghapus objek tidak dapat dibatalkan.
> - Tindakan ini menghapus semua objek yang telah ditentukan. Saat menghapus folder, tunggu hingga tindakan penghapusannya selesai sebelum menambahkan objek baru ke folder tersebut. Jika tidak, objek baru mungkin juga terhapus.

> **Note**
>
> Saat Anda menghapus beberapa objek secara terprogram dari bucket direktori, perhatikan hal berikut:
>
> - Kunci objek dalam permintaan `DeleteObjects` harus berisi setidaknya satu karakter spasi non-putih. String dari semua karakter spasi putih tidak didukung.
> - Kunci objek dalam `DeleteObjects` permintaan tidak dapat berisi karakter kontrol Unicode, kecuali untuk baris baru (`\n`), tab (`\t`), dan carriage return (`\r`).

Menggunakan konsol S3

Untuk menghapus objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih tab Bucket direktori.
4. Pilih bucket direktori yang berisi objek yang ingin Anda hapus.
5. Pilih tab Objek. Dalam daftar Objek, pilih kotak centang di sebelah kiri objek atau objek yang ingin Anda hapus.
6. Pilih Hapus.
7. Pada halaman Hapus objek, masukkan **`permanently delete`** di kotak teks.
8. Pilih Hapus objek.

Menggunakan AWS SDK

SDK for Java 2.x

Example

Contoh berikut menghapus objek dalam bucket direktori dengan menggunakan file. AWS SDK for Java 2.x

```
static void deleteObject(S3Client s3Client, String bucketName, String objectKey) {
```

```
        try {

            DeleteObjectRequest del = DeleteObjectRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .build();

            s3Client.deleteObject(del);

            System.out.println("Object " + objectKey + " has been deleted");


        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

    }
```

SDK for Python

Example

Contoh berikut menghapus objek dalam bucket direktori dengan menggunakan file. AWS SDK for Python (Boto3)

```
import logging
import boto3
from botocore.exceptions import ClientError

def delete_objects(s3_client, bucket_name, objects):
    '''
    Delete a list of objects in a directory bucket

    :param s3_client: boto3 S3 client
    :param bucket_name: Bucket that contains objects to be deleted; for example,
 'doc-example-bucket--usw2-az1--x-s3'
    :param objects: List of dictionaries that specify the key names to delete
    :return: Response output, else False
    '''
```

```
    try:
        response = s3_client.delete_objects(
            Bucket = bucket_name,
            Delete = {
                'Objects': objects
            }
        )
        return response
    except ClientError as e:
        logging.error(e)
        return False


if __name__ == '__main__':
    region = 'us-west-2'
    bucket_name = 'BUCKET_NAME'
    objects = [
        {
            'Key': '0.txt'
        },
        {
            'Key': '1.txt'
        },
        {
            'Key': '2.txt'
        },
        {
            'Key': '3.txt'
        },
        {
            'Key': '4.txt'
        }
    ]

    s3_client = boto3.client('s3', region_name = region)
    results = delete_objects(s3_client, bucket_name, objects)
    if results is not None:
        if 'Deleted' in results:
            print (f'Deleted {len(results["Deleted"])} objects from {bucket_name}')
        if 'Errors' in results:
            print (f'Failed to delete {len(results["Errors"])} objects from
 {bucket_name}')
```

Menggunakan AWS CLI

`delete-object`Contoh perintah berikut menunjukkan bagaimana Anda dapat menggunakan AWS CLI untuk menghapus objek dari bucket direktori. Untuk menjalankan perintah ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3api delete-object --bucket bucket-base-name--azid--x-s3 --key KEY_NAME
```

Untuk informasi selengkapnya, lihat delete-object dalam AWS CLI Referensi Perintah.

Mengunduh objek dalam ember direktori

Contoh kode berikut menunjukkan cara membaca data dari (mengunduh) objek di bucket direktori Amazon S3 dengan menggunakan operasi `GetObject` API.

Menggunakan AWS SDK

SDK for Java 2.x

Example

Contoh kode berikut menunjukkan cara membaca data dari objek dalam ember direktori dengan menggunakan AWS SDK for Java 2.x.

```
public static void getObject(S3Client s3Client, String bucketName, String objectKey)
 {
     try {
         GetObjectRequest objectRequest = GetObjectRequest
            .builder()
            .key(objectKey)
            .bucket(bucketName)
            .build();

         ResponseBytes GetObjectResponse objectBytes =
 s3Client.getObjectAsBytes(objectRequest);
         byte[] data = objectBytes.asByteArray();

         //Print object contents to console
         String s = new String(data, StandardCharsets.UTF_8);
         System.out.println(s);
    }
```

```
    catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

SDK for Python

Example

Contoh kode berikut menunjukkan cara membaca data dari objek dalam ember direktori dengan menggunakan AWS SDK for Python (Boto3).

```
import boto3
from botocore.exceptions import ClientError
from botocore.response import StreamingBody

def get_object(s3_client: boto3.client, bucket_name: str, key_name: str) ->
 StreamingBody:
    """
    Gets the object.
    :param s3_client:
    :param bucket_name: The bucket that contains the object.
    :param key_name: The key of the object to be downloaded.
    :return: The object data in bytes.
    """
    try:
        response = s3_client.get_object(Bucket=bucket_name, Key=key_name)
        body = response['Body'].read()
        print(f"Got object '{key_name}' from bucket '{bucket_name}'.")
    except ClientError:
        print(f"Couldn't get object '{key_name}' from bucket '{bucket_name}'.")
        raise
    else:
        return body

def main():
    s3_client = boto3.client('s3')
    resp = get_object(s3_client, 'doc-example-bucket--use1-az4--x-s3', 'sample.txt')
    print(resp)

if __name__ == "__main__":
```

```
    main()
```

Menggunakan AWS CLI

Contoh perintah `get-object` berikut menunjukkan bagaimana Anda dapat menggunakan AWS CLI untuk mengunduh objek dari Amazon S3. Perintah ini mendapatkan objek *KEY_NAME* dari bucket direktori*bucket-base-name*--*azid*--x-s3. Objek akan diunduh ke file yang bernama *LOCAL_FILE_NAME*. Untuk menjalankan perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3api get-object --bucket bucket-base-name--azid--x-s3 --
key KEY_NAME LOCAL_FILE_NAME
```

Untuk informasi selengkapnya, lihat get-object dalam AWS CLI Referensi Perintah.

Menggunakan **`HeadObject`** dengan ember direktori

Contoh AWS SDK dan AWS CLI berikut menunjukkan cara menggunakan `HeadObject` operasi API untuk mengambil metadata dari objek di bucket direktori Amazon S3 tanpa mengembalikan objek itu sendiri.

Menggunakan AWS SDK

SDK for Java 2.x

Example

```
public static void headObject(S3Client s3Client, String bucketName, String
 objectKey) {
    try {
        HeadObjectRequest headObjectRequest = HeadObjectRequest
                .builder()
                .bucket(bucketName)
                .key(objectKey)
                .build();
        HeadObjectResponse response = s3Client.headObject(headObjectRequest);
        System.out.format("Amazon S3 object: \"%s\" found in bucket: \"%s\" with
 ETag: \"%s\"", objectKey, bucketName, response.eTag());
    }
```

```
    catch (S3Exception e) {
        System.err.println(e.awsErrorDetails().errorMessage());
```

Menggunakan AWS CLI

`head-object`Contoh perintah berikut menunjukkan bagaimana Anda dapat menggunakan AWS CLI untuk mengambil metadata dari objek. Untuk menjalankan perintah ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3api head-object --bucket bucket-base-name--azid--x-s3 --key KEY_NAME
```

Untuk informasi selengkapnya, lihat head-object dalam AWS CLI Referensi Perintah.

Keamanan untuk S3 Express One Zone

Keamanan cloud di AWS merupakan prioritas tertinggi. Sebagai pelanggan AWS, Anda mendapatkan manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan dari organisasi yang paling sensitif terhadap keamanan. Keamanan adalah tanggung jawab bersama antara AWS dan Anda. Model tanggung jawab bersama menjelaskan hal ini sebagai keamanan cloud dan keamanan dalam cloud:

- Keamanan cloud – AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan Layanan AWS dalam AWS Cloud. AWS juga memberi Anda layanan yang dapat digunakan dengan aman. Auditor pihak ketiga melakukan pengujian dan verifikasi secara berkala terhadap efektivitas keamanan kami sebagai bagian dari AWS Compliance Programs.

  Untuk mempelajari program kepatuhan yang berlaku bagi Amazon S3 Express One Zone, lihat Layanan AWS in Scope by Compliance Program.
- Keamanan dalam cloud – Tanggung jawab Anda ditentukan oleh Layanan AWS yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, termasuk sensitivitas data Anda, persyaratan perusahaan Anda, serta hukum dan peraturan yang berlaku.

Dokumentasi ini akan membantu Anda memahami cara menerapkan model tanggung jawab bersama saat Anda menggunakan S3 Express One Zone. Topik berikut menunjukkan kepada Anda cara mengonfigurasi S3 Express One Zone untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga akan mempelajari cara menggunakan Layanan AWS lain yang dapat membantu Anda memantau dan mengamankan sumber daya Anda saat bekerja dengan S3 Express One Zone.

Topik

- Perlindungan data dan enkripsi
- AWS Identity and Access Management (IAM) untuk S3 Express One Zone
- Kebijakan berbasis identitas IAM untuk S3 Express One Zone
- Contoh kebijakan bucket direktori untuk S3 Express One Zone
- CreateSessionotorisasi
- Praktik terbaik keamanan untuk S3 Express One Zone

Perlindungan data dan enkripsi

Untuk informasi selengkapnya tentang cara S3 Express One Zone mengenkripsi dan melindungi data Anda, lihat topik berikut.

Topik

- Enkripsi Sisi Server dengan kunci terkelola Amazon S3 (SSE-S3)
- Enkripsi dalam bergerak
- Checksum tambahan
- Penghapusan data

Enkripsi Sisi Server dengan kunci terkelola Amazon S3 (SSE-S3)

Secara default, semua objek yang disimpan dalam bucket direktori secara otomatis dienkripsi dengan menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3). Unggahan yang tidak terenkripsi ke bucket direktori tidak diizinkan. Lihat informasi yang lebih lengkap di Menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) dan Melindungi data dengan enkripsi.

Bucket direktori tidak mendukung enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS), enkripsi sisi server dua lapis dengan kunci AWS Key Management Service (AWS KMS) (DSSE-KMS), atau enkripsi sisi server dengan kunci enkripsi yang disediakan pelanggan (SSE-C).

Enkripsi dalam bergerak

S3 Express One Zone hanya dapat diakses melalui HTTPS (TLS).

S3 Express One Zone menggunakan titik akhir API Regional dan Zonal. Bergantung pada operasi Amazon S3 API yang Anda gunakan, titik akhir Regional atau Zonal diperlukan. Anda dapat mengakses titik akhir Zonal dan Regional melalui cloud privat virtual (VPC) endpoint gateway. Tidak dikenakan biaya tambahan untuk menggunakan titik akhir gateway. Untuk mempelajari selengkapnya tentang titik akhir API Regional dan Zonal, lihat Jaringan untuk S3 Express One Zone.

Checksum tambahan

S3 Express One Zone menawarkan kepada Anda opsi untuk memilih algoritma checksum yang digunakan untuk memvalidasi data Anda selama mengunggah atau mengunduh. Anda dapat memilih salah satu algoritma pemeriksaan integritas data Secure Hash Algorithms (SHA) atau Cyclic Redundancy Check (CRC) berikut: CRC32, CRC32C, SHA-1, dan SHA-256. Checksum berbasis MD5 tidak didukung dengan kelas penyimpanan S3 Express One Zone.

Untuk informasi selengkapnya, lihat Praktik terbaik checksum tambahan S3.

Penghapusan data

Anda dapat menghapus satu atau beberapa objek langsung dari S3 Express One Zone dengan menggunakan konsol Amazon S3, SDK AWS, AWS Command Line Interface (AWS CLI), atau dengan API REST Amazon S3. Karena semua objek dalam direktori bucket Anda dikenakan biaya penyimpanan, sebaiknya hapus objek yang tidak diperlukan lagi.

Menghapus objek yang disimpan dalam sebuah bucket direktori juga secara rekursif menghapus setiap direktori induk, jika direktori-direktori induk tersebut tidak berisi objek lain selain objek yang sedang dihapus.

> **Note**
>
> Penghapusan autentikasi multi-faktor (MFA) dan Versioning S3 tidak didukung untuk S3 Express One Zone.

AWS Identity and Access Management (IAM) untuk S3 Express One Zone

AWS Identity and Access Management (IAM) adalah sebuah Layanan AWS yang membantu administrator mengontrol akses ke sumber daya dengan aman. AWS Administrator IAM mengendalikan siapa saja yang dapat diautentikasi (masuk) dan diizinkan (memiliki izin) untuk

menggunakan sumber daya Amazon S3 di S3 Express One Zone. Anda dapat menggunakan IAM tanpa biaya tambahan.

Secara default, pengguna tidak memiliki izin untuk bucket direktori dan operasi S3 Express One Zone. Untuk memberikan izin akses bagi bucket direktori, Anda dapat menggunakan IAM untuk membuat pengguna, grup, atau peran dan melampirkan izin ke identitas tersebut. Untuk informasi selengkapnya tentang IAM, lihat Praktik terbaik keamanan di IAM di Panduan Pengguna IAM.

Untuk memberikan akses, Anda dapat menambahkan izin ke pengguna, grup, atau peran Anda melalui cara berikut:

- Pengguna dan grup di AWS IAM Identity Center - Buat set izin. Ikuti instruksi di Buat rangkaian izin di Panduan Pengguna AWS IAM Identity Center .
- Pengguna yang dikelola di IAM melalui penyedia identitas - Buat peran untuk federasi identitas. Ikuti instruksi dalam Membuat peran untuk penyedia identitas pihak ketiga (federasi) di Panduan Pengguna IAM.
- Peran dan pengguna IAM – Buat peran yang dapat diambil oleh pengguna Anda. Ikuti instruksi dalam Membuat peran untuk mendelegasikan izin ke pengguna IAM di Panduan Pengguna IAM.

Secara default, bucket direktori bersifat pribadi dan hanya dapat diakses oleh pengguna yang secara eksplisit diberikan akses. Batas kontrol akses untuk bucket direktori hanya diatur pada tingkat bucket. Sebaliknya, batas kontrol akses untuk bucket tujuan umum dapat diatur pada tingkat bucket, prefiks, atau tag objek. Perbedaan ini berarti bucket direktori adalah satu-satunya sumber daya yang dapat Anda sertakan dalam kebijakan bucket atau kebijakan identitas IAM untuk akses S3 Express One Zone.

Dengan S3 Express One Zone, selain otorisasi IAM, Anda mengautentikasi dan mengotorisasi permintaan melalui mekanisme berbasis sesi baru yang ditangani oleh operasi API `CreateSession`. Anda dapat menggunakan `CreateSession` untuk meminta kredensial sementara yang menyediakan akses latensi rendah ke bucket Anda. Kredensial sementara ini dicakup ke bucket direktori tertentu.

Untuk bekerja dengan`CreateSession`, kami sarankan menggunakan versi terbaru dari AWS SDK atau menggunakan AWS Command Line Interface (AWS CLI). AWS SDK yang didukung dan pembentukan sesi AWS CLI penanganan, penyegaran, dan penghentian atas nama Anda.

Anda menggunakan token sesi dengan operasi Zona (tingkat objek) saja (kecuali untuk `CopyObject` dan`HeadBucket`) untuk mendistribusikan latensi yang terkait dengan otorisasi atas sejumlah

permintaan dalam sesi. Untuk operasi API titik akhir Regional (operasi tingkat bucket), Anda menggunakan otorisasi IAM, yang tidak melibatkan pengelolaan sesi. Lihat informasi yang lebih lengkap di AWS Identity and Access Management (IAM) untuk S3 Express One Zone dan `CreateSession`otorisasi.

Untuk informasi tentang pengaturan IAM untuk S3 Express One Zone, lihat topik berikut.

Topik

- Pengguna utama
- Sumber daya
- Tindakan untuk S3 Express One Zone
- Kunci syarat untuk S3 Express One Zone
- Bagaimana operasi API diotorisasi dan diautentikasi

Pengguna utama

Saat membuat kebijakan berbasis sumber daya untuk memberikan akses ke bucket, Anda harus menggunakan elemen `Principal` tersebut untuk menentukan orang atau aplikasi yang dapat membuat permintaan tindakan atau operasi pada sumber daya tersebut. Untuk kebijakan bucket direktori, Anda dapat menggunakan pengguna utama berikut:

- AWS Akun
- Pengguna IAM
- Peran IAM
- Pengguna gabungan

Untuk informasi selengkapnya, lihat Principal dalam Panduan Pengguna IAM.

Sumber daya

Nama Sumber Daya Amazon (ARN) untuk bucket direktori berisi `s3express` namespace, ID AWS akun Wilayah AWS, dan nama bucket direktori, yang menyertakan ID Availability Zone. Untuk mengakses dan melakukan tindakan pada bucket direktori Anda, Anda harus menggunakan format ARN berikut ini:

```
arn:aws:s3express:region:account-id:bucket/base-bucket-name--azid--x-s3
```

Untuk informasi selengkapnya tentang ARN, lihat Amazon Resource Names (ARNs) dalam Panduan Pengguna IAM. Untuk informasi selengkapnya tentang sumber daya, lihat Elemen Kebijakan JSON IAM: Resource dalam Panduan Pengguna IAM.

Tindakan untuk S3 Express One Zone

Dalam kebijakan berbasis identitas atau kebijakan berbasis sumber daya IAM, Anda menentukan tindakan S3 yang diizinkan atau ditolak. Tindakan S3 Express One Zone sesuai dengan operasi API tertentu. S3 Express One Zone memiliki namespace IAM unik yang berbeda dari namespace standar untuk Amazon S3. Namespace ini adalah `s3express`.

Saat Anda memberikan izin `s3express:CreateSession`, ini memungkinkan operasi API `CreateSession` untuk mengambil token sesi saat mengakses operasi API (atau tingkat objek) titik akhir Zona. Token sesi ini mengembalikan kredensial yang digunakan untuk memberikan akses ke semua operasi API titik akhir Zona lainnya. Akibatnya, Anda tidak perlu memberikan izin akses ke operasi API Zona menggunakan kebijakan IAM. Sebagai gantinya, token sesi memungkinkan akses.

Untuk informasi selengkapnya tentang operasi API titik akhir Zona dan Regional, lihat Jaringan untuk S3 Express One Zone. Untuk mempelajari lebih lanjut tentang operasi API `CreateSession`, lihat CreateSession di Referensi API Amazon Simple Storage Service.

Anda dapat menyebutkan tindakan berikut dalam elemen `Action` pernyataan kebijakan IAM. Gunakan kebijakan untuk memberikan izin untuk melaksanakan operasi dalam AWS. Saat Anda menggunakan tindakan dalam kebijakan, Anda biasanya mengizinkan atau menolak akses ke operasi API dengan nama yang sama. Namun, dalam beberapa kasus, satu tindakan mengendalikan akses ke lebih dari satu operasi API. Akses ke tindakan tingkat bucket dapat diberikan hanya dalam kebijakan berbasis identitas IAM (pengguna atau peran) dan bukan kebijakan bucket.

Tindakan dan kunci syarat untuk S3 Express One Zone

Tindakan	API	Deskripsi	Tingkat akses	Kunci syarat
`s3express:CreateBucket`	`CreateBucket`	Memberikan izin untuk membuat bucket baru.	Tulis	`s3express:authType` `s3express:LocationName`

Tindakan	API	Deskripsi	Tingkat akses	Kunci syarat

				`s3express:ResourceAccount`
				`s3express:signatureversion`
				`s3express:TlsVersion`
				`s3express:x-amz-content-sha256`

Tindakan	API	Deskripsi	Tingkat akses	Kunci syarat

`s3express:CreateSession`	`CreateSession`	Memberikan izin untuk membuat token sesi, yang digunakan untuk memberikan akses ke semua operasi API Zona (tingkat objek), seperti `PutObject` , `GetObject` , dan sebagainya.	Tulis	`s3express:authType`
				
				`s3express:SessionMode`
				
				`s3express:ResourceAccount`
				
				`s3express:signaturversion`
				
				`s3express:signatureAge`
				
				`s3express:TlsVersion`
				
				`s3express:x-amz-content-sha256`

Tindakan	API	Deskripsi	Tingkat akses	Kunci syarat

`s3express:DeleteBucket`	`DeleteBucket`	Memberikan izin untuk menghapus bucket yang namanya tercantum dalam URI.	Tulis	`s3express:authType`
				`s3express:ResourceAccount`
				`s3express:signatureversion`
				`s3express:TlsVersion`
				`s3express:x-amz-content-sha256`

Tindakan	API	Deskripsi	Tingkat akses	Kunci syarat

`s3express:DeleteBucketPolicy`	`DeleteBucketPolicy`	Memberikan izin untuk menghapus kebijakan pada bucket yang ditentukan.	Manajem izin	`s3express:authType`
				`s3express:ResourceAccount`
				`s3express:signatureversion`
				`s3express:TlsVersion`
				`s3express:x-amz-content-sha256`

Tindakan	API	Deskripsi	Tingkat akses	Kunci syarat

`s3express:GetBucketPolicy`	`GetBucketPolicy`	Memberikan izin untuk mengembalikan kebijakan dari bucket yang ditentukan.	Baca	`s3express:authType`
				`s3express:ResourceAccount`
				`s3express:signatureversion`
				`s3express:TlsVersion`
				`s3express:x-amz-content-sha256`

Tindakan	API	Deskripsi	Tingkat akses	Kunci syarat

`s3express:ListAllMyDirectoryBuckets`	`ListDirectoryBuckets`	Memberikan izin untuk mencantumkan semua bucket direktori yang dimiliki oleh pengirim permintaan yang diautentikasi.	Daftar	`s3express:authType`
				`s3express:ResourceAccount`
				`s3express:signatureversion`
				`s3express:TlsVersion`
				`s3express:x-amz-content-sha256`

Tindakan	API	Deskripsi	Tingkat akses	Kunci syarat

`s3express:PutBucketPolicy`	`PutBucketPolicy`	Memberikan izin untuk menambah atau mengganti kebijakan bucket pada bucket.	Manajem izin	`s3express:authType`
				`s3express:ResourceAccount`
				`s3express:signaturversion`
				`s3express:TlsVersion`
				`s3express:x-amz-content-sha256`

Kunci syarat untuk S3 Express One Zone

S3 Express One Zone mendefinisikan kunci syarat berikut yang dapat digunakan dalam elemen `Condition` kebijakan IAM. Anda dapat menggunakan kunci ini untuk menyempurnakan syarat lebih lanjut saat pernyataan kebijakan berlaku.

Kunci syarat	Deskripsi	Jenis
`s3express:authType`	Filter akses berdasarkan metode autentikasi. Untuk membatasi permintaan masuk untuk menggunakan metode autentikasi tertentu, Anda dapat menggunakan kunci syarat opsional ini. Misalnya, Anda dapat menggunakan kunci syarat ini untuk mengizinkan hanya header	String

Kunci syarat	Deskripsi	Jenis
	`Authorization` HTTP yang digunakan dalam autentikasi permintaan. Nilai yang valid: `REST-HEADER` , `REST-QUERY-STRING`	
`s3express:LocationName`	Memfilter akses ke operasi API `CreateBucket` dengan ID Zona Ketersediaan tertentu (ID AZ), misalnya, `usw2-az1`. Nilai contoh: `usw2-az1`	String
`s3express:ResourceAccount`	Memfilter akses oleh Akun AWS ID pemilik sumber daya. Untuk membatasi akses pengguna, peran, atau aplikasi ke bucket direktori yang dimiliki oleh Akun AWS ID tertentu, Anda dapat menggunakan kunci `aws:ResourceAccount` atau `s3express:ResourceAccount` kondisi. Anda dapat menggunakan kunci kondisi ini dalam kebijakan identitas AWS Identity and Access Management (IAM) atau kebijakan titik akhir virtual private cloud (VPC). Misalnya, Anda dapat menggunakan kunci kondisi ini untuk membatasi klien dalam VPC Anda agar tidak mengakses bucket yang tidak Anda miliki. Nilai contoh: `111122223333`	String

Kunci syarat	Deskripsi	Jenis
`s3express:SessionMode`	Memfilter akses berdasarkan izin yang diminta oleh operasi API `CreateSession` . Secara default, sesi adalah `ReadWrite` . Anda dapat menggunakan kunci syarat ini untuk membatasi akses ke `ReadOnly` atau secara eksplisit menolak akses `ReadWrite` . Untuk informasi selengkapnya, lihat Contoh kebijakan bucket direktori untuk S3 Express One Zone dan CreateSession dalam Referensi API Amazon Simple Storage Service. Nilai yang valid: `ReadWrite` , `ReadOnly`	String
`s3express:signatureAge`	Filter akses berdasarkan umur tanda tangan permintaan dalam milidetik. Syarat ini hanya berfungsi untuk URL yang telah ditandatangani sebelumnya. Di AWS Signature Version 4, kunci penandatanganan berlaku hingga tujuh hari. Oleh karena itu, tanda tangan juga berlaku hingga tujuh hari. Untuk informasi selengkapnya, lihat Pendahuluan ke permintaan penandatanganan dalam Referensi API Amazon Simple Storage Service. Anda dapat menggunakan syarat ini untuk lebih lanjut membatasi umur tanda tangan. Nilai contoh: `600000`	Numerik

Kunci syarat	Deskripsi	Jenis
`s3express:signatureversion`	Mengidentifikasi versi AWS Signature yang ingin Anda dukung untuk permintaan yang diautentikasi. Untuk permintaan yang diautentikasi, S3 Express One Zone mendukung Signature Version 4. Nilai valid: `"AWS4-HMAC-SHA256"` (mengidentifikasi Signature Version 4)	String
`s3express:TlsVersion`	Filter akses berdasarkan versi TLS yang digunakan oleh klien. Anda dapat menggunakan kunci `s3:TlsVersion` kondisi untuk menulis IAM, virtual private cloud endpoint (VPCE), atau kebijakan bucket yang membatasi akses pengguna atau aplikasi ke bucket direktori berdasarkan versi TLS yang digunakan oleh klien. Anda juga dapat menggunakan kunci syarat ini untuk menulis kebijakan yang memerlukan versi TLS minimum. Nilai contoh: `1.3`	Numerik

Kunci syarat	Deskripsi	Jenis
`s3express:x-amz-content-sha256`	Filter akses berdasarkan konten yang belum ditandatangani di bucket Anda. Anda dapat menggunakan kunci syarat ini untuk melarang konten yang tidak ditandatangani di bucket Anda. Ketika Anda menggunakan Signature Version 4 untuk permintaan yang menggunakan header `Authorization` , Anda menambahkan header `x-amz-content-sha256` dalam perhitungan tanda tangan, lalu kemudian menetapkan nilainya ke muatan hash. Anda dapat menggunakan kunci syarat ini dalam kebijakan bucket untuk menolak setiap unggahan yang muatannya tidak ditandatangani. Misalnya: • Menolak unggahan yang menggunakan header `Authorization` untuk mengautentikasi permintaan tetapi tidak menandatangani muatan. Untuk informasi selengkapnya, lihat Memindahkan muatan dalam satu bagian tunggal di Referensi API Amazon Simple Storage Service. • Tolak unggahan yang menggunakan URL yang telah ditetapkan sebelumnya. URL yang telah ditandatangani sebelumnya selalu memiliki `UNSIGNED_PAYLOAD` . Untuk informasi selengkapnya, lihat Mengautentikasi permintaan dan Metode autentikasi dalam Referensi API Amazon Simple Storage Service.	String

Kunci syarat	Deskripsi	Jenis
	Nilai yang valid: `UNSIGNED-PAYLOAD`	

Bagaimana operasi API diotorisasi dan diautentikasi

Tabel berikut mencantumkan informasi otorisasi dan autentikasi untuk operasi API S3 Express One Zone. Untuk setiap operasi API, tabel menunjukkan nama operasi API, tindakan IAM, jenis titik akhir (Regional atau Zona), dan mekanisme otorisasi (IAM atau berbasis sesi). Tabel ini juga menunjukkan tempat akses lintas akun didukung. Akses ke tindakan tingkat bucket dapat diberikan hanya dalam kebijakan berbasis identitas IAM (pengguna atau peran), bukan kebijakan bucket.

API	Jenis titik akhir	Tindakan IAM	Akses lintas akun
`CreateBucket`	Regional	`s3express:CreateBucket`	Tidak
`DeleteBucket`	Regional	`s3express:DeleteBucket`	Tidak
`ListDirectoryBuckets`	Regional	`s3express:ListAllMyDirectoryBuckets`	Tidak
`PutBucketPolicy`	Regional	`s3express:PutBucketPolicy`	Tidak
`GetBucketPolicy`	Regional	`s3express:GetBucketPolicy`	Tidak
`DeleteBucketPolicy`	Regional	`s3express:DeleteBucketPolicy`	Tidak
`CreateSession`	Zona	`s3express:CreateSession`	Ya
`CopyObject`	Zona	`s3express:CreateSession`	Ya
`DeleteObject`	Zona	`s3express:CreateSession`	Ya
`DeleteObjects`	Zona	`s3express:CreateSession`	Ya

API	Jenis titik akhir	Tindakan IAM	Akses lintas akun
HeadObject	Zona	s3express:CreateSession	Ya
PutObject	Zona	s3express:CreateSession	Ya
GetObjectAttributes	Zona	s3express:CreateSession	Ya
ListObjectsV2	Zona	s3express:CreateSession	Ya
HeadBucket	Zona	s3express:CreateSession	Ya
CreateMultipartUpload	Zona	s3express:CreateSession	Ya
UploadPart	Zona	s3express:CreateSession	Ya
UploadPartCopy	Zona	s3express:CreateSession	Ya
CompleteMultipartUpload	Zona	s3express:CreateSession	Ya
AbortMultipartUpload	Zona	s3express:CreateSession	Ya
ListParts	Zona	s3express:CreateSession	Ya
ListMultipartUploads	Zona	s3express:CreateSession	Ya

Kebijakan berbasis identitas IAM untuk S3 Express One Zone

Sebelum Anda dapat membuat bucket direktori atau menggunakan kelas penyimpanan Amazon S3 Express One Zone, Anda harus memberikan izin yang diperlukan untuk peran atau pengguna AWS Identity and Access Management (IAM) Anda. Kebijakan contoh ini memungkinkan akses

ke operasi API `CreateSession` (untuk digunakan dengan operasi API titik akhir [tingkat objek] Zona) dan semua operasi API titik akhir Regional (tingkat bucket). Kebijakan ini memungkinkan operasi API `CreateSession` untuk digunakan dengan semua bucket direktori, tetapi operasi API titik akhir Regional hanya diizinkan untuk digunakan dengan bucket direktori yang ditentukan. Untuk menggunakan kebijakan contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
{
     "Version": "2012-10-17",
     "Statement": [
         {
             "Sid": "AllowAccessRegionalEndpointAPIs",
             "Effect": "Allow",
             "Action": [
                 "s3express:DeleteBucket",
                 "s3express:DeleteBucketPolicy",
                 "s3express:CreateBucket",
                 "s3express:PutBucketPolicy",
                 "s3express:GetBucketPolicy",
                 "s3express:ListAllMyDirectoryBuckets"
             ],

             "Resource": "arn:aws:s3express:region:account_id:bucket/bucket-base-
name--azid--x-s3/*"
         },
         {
             "Sid": "AllowCreateSession",
             "Effect": "Allow",
             "Action": "s3express:CreateSession",
             "Resource": "*"
         }
     ]
 }
```

Contoh kebijakan bucket direktori untuk S3 Express One Zone

Bagian ini menyediakan contoh kebijakan bucket direktori untuk digunakan dengan kelas penyimpanan Amazon S3 Express One Zone. Untuk menggunakan kebijakan ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

Contoh kebijakan bucket berikut memungkinkan ID Akun AWS *111122223333* untuk menggunakan operasi API `CreateSession` dengan sesi `ReadWrite` default untuk bucket direktori yang ditentukan. Kebijakan ini memberikan akses ke operasi API titik akhir Zona (tingkat objek).

Example – Kebijakan bucket untuk mengizinkan panggilan **`CreateSession`** dengan sesi **`ReadWrite`** default

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteAccess",
            "Effect": "Allow",
            "Resource": "arn:aws:s3express:us-west-2:account-id:bucket/bucket-base-
name--azid--x-s3",
            "Principal": {
                "AWS": [
                    "111122223333"
                ]
            },
            "Action": [
                "s3express:CreateSession"
            ]
        }
    ]
}
```

Example – Kebijakan bucket untuk mengizinkan panggilan **`CreateSession`** dengan sesi **`ReadOnly`**

Contoh kebijakan bucket berikut memungkinkan ID Akun AWS *111122223333* untuk menggunakan operasi API `CreateSession`. Kebijakan ini menggunakan kunci syarat `s3express:SessionMode` dengan nilai `ReadOnly` untuk mengatur sesi hanya baca.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadOnlyAccess",
            "Effect": "Allow",
```

```
            "Principal": {
                "AWS": "111122223333"
            },
            "Action": "s3express:CreateSession",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3express:SessionMode": "ReadOnly"
                }
            }
        }
    ]
}
```

Example – Kebijakan bucket untuk memungkinkan akses lintas akun bagi panggilan **`CreateSession`**

Contoh kebijakan bucket berikut memungkinkan ID Akun AWS *111122223333* untuk menggunakan operasi API `CreateSession` bagi bucket direktori yang ditentukan yang dimiliki oleh ID Akun AWS*444455556666*.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CrossAccount",
            "Effect": "Allow",
            "Principal": {
                "AWS": "111122223333"
            },
            "Action": [
                "s3express:CreateSession"
            ],
            "Resource": "arn:aws:s3express:us-west-2:444455556666:bucket/bucket-base-
name--azid--x-s3"

        }
    ]
}
```

`CreateSession`otorisasi

Amazon S3 Express One Zone mendukung otorisasi AWS Identity and Access Management (AWSIAM) dan otorisasi berbasis sesi:

- Untuk menggunakan operasi API titik akhir Regional (operasi tingkat ember, atau bidang kontrol) dengan S3 Express One Zone, Anda menggunakan model otorisasi IAM, yang tidak melibatkan manajemen sesi. Izin diberikan untuk tindakan yang dilakukan secara individual. Untuk informasi selengkapnya, lihat AWS Identity and Access Management (IAM) untuk S3 Express One Zone.
- Untuk menggunakan operasi API titik akhir Zonal (operasi tingkat objek, atau bidang data), gunakan operasi `CreateSession` API untuk membuat dan mengelola sesi yang dioptimalkan untuk otorisasi permintaan data latensi rendah. Untuk mengambil dan menggunakan token sesi, Anda harus mengizinkan tindakan `s3express:CreateSession` untuk bucket direktori Anda dalam kebijakan berbasis identitas atau kebijakan bucket. Untuk informasi selengkapnya, lihat AWS Identity and Access Management (IAM) untuk S3 Express One Zone. Jika Anda mengakses S3 Express One Zone di konsol Amazon S3, melalui AWS Command Line Interface (AWS CLI), atau dengan menggunakan SDK AWS, S3 Express One Zone membuat sesi atas nama Anda.

Jika Anda menggunakan API REST Amazon S3, Anda dapat menggunakan operasi API `CreateSession` untuk mendapatkan kredensial keamanan sementara yang menyertakan ID kunci akses, kunci akses rahasia, token sesi, dan waktu kedaluwarsa. Kredensial sementara memberikan izin yang sama seperti kredensial keamanan jangka panjang, seperti kredensial pengguna IAM, tetapi kredensial keamanan sementara harus menyertakan token sesi.

Mode Sesi

Mode sesi mendefinisikan ruang lingkup sesi. Dalam kebijakan bucket, Anda dapat menentukan kunci kondisi `s3express:SessionMode` untuk mengontrol siapa yang dapat membuat sesi `ReadWrite` atau `ReadOnly`. Untuk informasi selengkapnya tentang sesi `ReadWrite` atau `ReadOnly`, lihat parameter `x-amz-create-session-mode` untuk CreateSession di Referensi API Amazon S3. Untuk informasi selengkapnya tentang kebijakan bucket yang akan dibuat, lihatContoh kebijakan bucket direktori untuk S3 Express One Zone.

Token Sesi

Saat Anda melakukan panggilan dengan menggunakan kredensial keamanan sementara, panggilan harus menyertakan token sesi. Token sesi dikembalikan bersama dengan kredensial sementara. Token sesi dibatasi pada bucket direktori Anda dan digunakan untuk memverifikasi bahwa kredensial

keamanan valid dan belum kedaluwarsa. Untuk melindungi sesi Anda, kredensial keamanan sementara akan kedaluwarsa setelah 5 menit.

`CopyObject` dan `HeadBucket`

Kredensial keamanan sementara dibatasi pada bucket direktori tertentu dan secara otomatis diaktifkan untuk semua panggilan API operasi Zonal (tingkat objek) ke bucket direktori yang diberikan. Tidak seperti operasi API titik akhir Zonal lainnya, `CopyObject` dan `HeadBucket` tidak menggunakan otentikasi `CreateSession`. Semua permintaan `CopyObject` dan `HeadBucket` harus diautentikasi dan ditandatangani dengan menggunakan kredensial IAM. Namun, `CopyObject` dan masih `HeadBucket` diotorisasi oleh `s3express:CreateSession`, seperti operasi API titik akhir Zonal lainnya.

Untuk informasi selengkapnya, lihat CreateSession dalam Referensi API Amazon Simple Storage Service.

Praktik terbaik keamanan untuk S3 Express One Zone

Amazon S3 Express One Zone menyediakan sejumlah fitur keamanan untuk dipertimbangkan ketika Anda mengembangkan dan menerapkan kebijakan keamanan Anda sendiri. Praktik terbaik berikut adalah pedoman umum dan tidak mewakili solusi keamanan yang lengkap. Karena praktik terbaik ini mungkin tidak sesuai atau memadai untuk lingkungan Anda, perlakukanlah sebagai rekomendasi yang bermanfaat, bukan sebagai resep.

Pengaturan Blokir Akses Publik dan Kepemilikan Objek Default

Untuk menggunakan kelas penyimpanan S3 Express One Zone, Anda harus menggunakan bucket direktori S3. Bucket direktori mendukung Blokir Akses Publik S3 dan Kepemilikan Objek S3. Fitur S3 ini digunakan untuk mengaudit dan mengelola akses ke bucket dan objek Anda.

Secara default, semua pengaturan Blok Akses Publik untuk bucket direktori diaktifkan. Selain itu, Kepemilikan Objek diatur ke mengaktifkan pemilik bucket, yang berarti daftar kontrol akses (ACL) dinonaktifkan. Pengaturan ini tidak dapat dimodifikasi. Untuk informasi selengkapnya tentang fitur ini, lihat the section called “Memblokir akses publik” dan the section called “Mengontrol kepemilikan objek”.

> **Note**
>
> Anda tidak dapat memberikan akses ke objek yang disimpan dalam bucket direktori. Anda hanya dapat memberikan akses ke bucket direktori Anda. Model otorisasi untuk S3 Express

One Zone berbeda dari model otorisasi untuk Amazon S3. Untuk informasi selengkapnya, lihat `CreateSession`otorisasi.

Autentikasi dan otorisasi

Mekanisme autentikasi dan otorisasi untuk S3 Express One Zone berbeda, tergantung pada jika Anda membuat permintaan ke operasi API titik akhir Zona atau operasi API titik akhir Regional. Operasi API Zona adalah operasi tingkat objek (bidang data). Operasi API regional adalah operasi tingkat bucket (bidang kontrol).

Dengan S3 Express One Zone, Anda mengautentikasi dan mengotorisasi permintaan ke operasi API titik akhir Zona melalui mekanisme berbasis sesi baru yang dioptimalkan untuk memberikan latensi terendah. Dengan autentikasi berbasis sesi, SDK AWS menggunakan operasi API `CreateSession` untuk meminta kredensial sementara yang menyediakan akses latensi rendah ke bucket direktori Anda. Kredensial sementara ini dicakup ke bucket direktori tertentu dan kedaluwarsa setelah 5 menit. Anda dapat menggunakan kredensial sementara ini untuk menandatangani panggilan API Zona (tingkat objek). Untuk informasi selengkapnya, lihat `CreateSession`otorisasi.

Menandatangani permintaan dengan kredensial S3 Express One Zone

Anda menggunakan kredensial S3 Express One Zone untuk menandatangani permintaan API titik akhir Zona (tingkat objek) dengan AWS Signature Version 4, dengan `s3express` sebagai nama layanan. Saat Anda menandatangani permintaan, gunakan kunci rahasia yang dikembalikan dari `CreateSession` dan berikan juga token sesi dengan `x-amzn-s3session-token header`. Untuk informasi selengkapnya, lihat CreateSession.

SDK AWS yang didukung untuk kelas S3 Express One Zone mengelola kredensial dan penandatanganan atas nama Anda. Kami menyarankan untuk menggunakan AWS SDK bagi S3 Express One Zone untuk memuat ulang kredensial dan menandatangani permintaan untuk Anda.

Menandatangani permintaan dengan kredensial IAM

Semua panggilan API Regional (tingkat bucket) harus diautentikasi dan ditandatangani oleh kredensial AWS Identity and Access Management (IAM), bukan kredensial sesi sementara. Kredensial IAM terdiri atas ID kunci akses dan kunci akses rahasia untuk identitas IAM. Semua permintaan `CopyObject` dan `HeadBucket` juga harus diautentikasi dan ditandatangani menggunakan kredensial IAM.

Untuk mencapai latensi terendah bagi panggilan operasi Zona (tingkat objek) Anda, kami menyarankan untuk menggunakan kredensial S3 Express One Zone yang diperoleh dari panggilan `CreateSession` untuk menandatangani permintaan Anda, kecuali untuk permintaan ke `CopyObject` dan `HeadBucket`.

Gunakan AWS CloudTrail

AWS CloudTrail memberikan catatan tindakan yang diambil oleh pengguna, peran, atau Layanan AWS di Amazon S3. Anda dapat menggunakan informasi yang dikumpulkan oleh CloudTrail untuk menentukan hal-hal berikut:

- Permintaan yang diajukan ke Amazon S3
- Alamat IP dari mana permintaan itu dibuat
- Siapa yang membuat permintaan
- Saat permintaan dibuat
- Detail tambahan tentang permintaan

Saat Anda mengaturAkun AWS, CloudTrail diaktifkan secara default. Operasi API titik akhir Regional berikut (tingkat ember, atau bidang kontrol, operasi API) dicatat. CloudTrail

- `CreateBucket`
- `DeleteBucket`
- `DeleteBucketPolicy`
- `PutBucketPolicy`
- `GetBucketPolicy`
- `ListDirectoryBuckets`

Anda dapat melihat peristiwa terbaru di CloudTrail konsol. Untuk membuat catatan aktivitas dan peristiwa yang sedang berlangsung untuk bucket Amazon S3, Anda dapat membuat jejak di konsol. CloudTrail Untuk informasi selengkapnya, lihat Creating a trail di AWS CloudTrail Panduan Pengguna.

> **Note**
>
> Untuk S3 Express One Zone, CloudTrail pencatatan operasi API titik akhir Zonal (level objek, atau bidang data) (misalnya, `PutObject` atau`GetObject`) tidak didukung.

Terapkan pemantauan menggunakan alat bantu pemantauan AWS

Pemantauan adalah bagian penting dari pemeliharaan keandalan, keamanan, ketersediaan, dan performa Amazon S3 dan solusi AWS Anda. AWS menyediakan beberapa alat dan layanan untuk membantu Anda memantau Amazon S3 dan layanan Layanan AWS lainnya. Misalnya, Anda dapat memantau CloudWatch metrik Amazon untuk Amazon S3, khususnya metrik penyimpanan `NumberOfObjects` dan `BucketSizeBytes` penyimpanan.

Objek yang disimpan di kelas penyimpanan S3 Express One Zone tidak akan tercermin dalam metrik penyimpanan `BucketSizeBytes` dan `NumberOfObjects` untuk Amazon S3. Namun, metrik penyimpanan `BucketSizeBytes` dan `NumberOfObjects` didukung untuk S3 Express One Zone. Untuk melihat metrik pilihan Anda, Anda dapat membedakan antara kelas penyimpanan Amazon S3 dan kelas penyimpanan S3 Express One Zone dengan menentukan dimensi `StorageType`. Untuk informasi selengkapnya, lihat Memantau metrik dengan Amazon CloudWatch.

Lihat informasi yang lebih lengkap di Memantau metrik dengan Amazon CloudWatch dan Pemantauan Amazon S3.

Mengoptimalkan Kinerja Amazon S3 Express One Zone

Amazon S3 Express One Zone adalah kelas penyimpanan S3 Zona Ketersediaan (AZ) tunggal berkinerja tinggi yang dibuat khusus untuk menghadirkan akses data milidetik satu digit yang konsisten untuk aplikasi Anda yang paling sensitif terhadap latensi. S3 Express One Zone adalah kelas penyimpanan S3 pertama yang memberi Anda opsi untuk mencari sumber daya penyimpanan objek dan komputasi AWS berkinerja tinggi, seperti Amazon Elastic Compute Cloud, Amazon Elastic Kubernetes Service, dan Amazon Elastic Container Service, dalam satu Zona Ketersediaan. Lokasi bersama penyimpanan dan sumber daya komputasi Anda mengoptimalkan kinerja dan biaya komputasi serta memberikan peningkatan kecepatan pemrosesan data.

S3 Express One Zone memberikan elastisitas kinerja yang serupa dengan kelas penyimpanan S3 lainnya, tetapi dengan latensi permintaan baca dan tulis satu digit milidetik yang konsisten—hingga 10x lebih cepat daripada Standar S3. S3 Express One Zone dirancang dari bawah ke atas

untuk mendukung throughput burst hingga level agregat yang sangat tinggi. Kelas penyimpanan S3 Express One Zone menggunakan arsitektur yang dibuat khusus untuk mengoptimalkan kinerja dan memberikan latensi permintaan rendah secara konsisten dengan menyimpan data pada perangkat keras berkinerja tinggi. Protokol objek untuk S3 Express One Zone telah ditingkatkan untuk melancarkan autentikasi dan overhead metadata.

Untuk lebih meningkatkan kecepatan akses dan mendukung ratusan ribu permintaan per detik, S3 Express One Zone menyimpan data dalam jenis bucket baru—bucket direktori Amazon S3. Setiap bucket direktori S3 dapat mendukung ratusan ribu transaksi per detik (TPS).

Kombinasi perangkat keras dan perangkat lunak berkinerja tinggi yang dibuat khusus yang memberikan kecepatan akses data milidetik satu digit dan bucket direktori yang diskalakan untuk sejumlah besar transaksi per detik menjadikan S3 Express One Zone sebagai kelas penyimpanan Amazon S3 terbaik untuk operasi intensif permintaan atau aplikasi yang kritis kinerja.

Topik berikut menjelaskan panduan praktik terbaik dan pola desain untuk mengoptimalkan kinerja aplikasi yang menggunakan kelas penyimpanan S3 Express One Zone.

Topik

- Pedoman kinerja dan pola desain untuk S3 Express One Zone

Pedoman kinerja dan pola desain untuk S3 Express One Zone

Saat membangun aplikasi yang mengunggah dan mengambil objek dari Amazon S3 Express One Zone, ikuti panduan praktik terbaik kami untuk mengoptimalkan kinerja. Untuk menggunakan kelas penyimpanan S3 Express One Zone, Anda harus membuat bucket direktori S3. Kelas penyimpanan S3 Express One Zone tidak didukung untuk digunakan dengan bucket tujuan umum S3.

Untuk panduan kinerja semua kelas penyimpanan Amazon S3 lainnya dan bucket tujuan umum S3, lihat Pola desain praktik terbaik: mengoptimalkan performa Amazon S3.

Untuk mendapatkan kinerja terbaik bagi aplikasi Anda saat menggunakan kelas penyimpanan dan bucket direktori S3 Express One Zone, kami merekomendasikan panduan dan pola desain berikut ini.

Topik

- Lokasi bersama penyimpanan S3 Express One Zone dengan sumber daya komputasi AWS Anda
- Bucket direktori
- Paralelisasi permintaan penskalaan horizontal bucket direktori

- Gunakan autentikasi berbasis sesi
- Praktik terbaik checksum tambahan S3
- Gunakan versi terbaru dari SDK AWS dan pustaka runtime umum
- Penyelesaian masalah kinerja

Lokasi bersama penyimpanan S3 Express One Zone dengan sumber daya komputasi AWS Anda

Setiap bucket direktori disimpan dalam satu Zona Ketersediaan yang Anda pilih saat membuat bucket. Anda dapat memulai dengan membuat bucket direktori baru di Zona Ketersediaan lokal untuk beban kerja atau sumber daya komputasi Anda. Anda kemudian dapat segera memulai membaca dan menulis latensi sangat rendah. Bucket direktori adalah bucket S3 pertama tempat Anda dapat memilih Zona Ketersediaan di Wilayah AWS untuk mengurangi latensi antara komputasi dan penyimpanan.

Apabila Anda mengakses bucket direktori di Zona Ketersediaan, latensi akan meningkat. Untuk mengoptimalkan performa, kami menyarankan agar Anda sebisa mungkin mengakses bucket direktori dari instans Amazon Elastic Container Service, Amazon Elastic Kubernetes Service, dan Amazon Elastic Compute Cloud yang berada di Zona Ketersediaan yang sama.

Bucket direktori

Setiap bucket direktori dapat mendukung ratusan ribu transaksi per detik (TPS). Tidak seperti bucket tujuan umum, bucket direktori mengatur kunci secara hierarkis ke dalam direktori alih-alih prefiks. Prefiks adalah string karakter di awal nama kunci objek. Anda dapat menganggap prefiks sebagai cara untuk mengatur data Anda dengan cara yang mirip dengan direktori. Namun, prefiks bukan direktori.

Prefiks mengatur data dalam namespace datar dalam bucket tujuan umum, dan tidak ada batas jumlah prefiks dalam bucket tujuan umum. Setiap awalan dapat mencapai setidaknya 3.500 `PUT``POST`/`DELETE`atau `HEAD` 5.500/permintaan per `GET` detik. Anda juga dapat memparalelkan permintaan di beberapa prefiks untuk menskalakan kinerja. Namun, penskalaan ini, dalam kasus operasi baca dan tulis, terjadi secara bertahap dan tidak instan. Selagi bucket tujuan umum menskalakan ke tingkat permintaan baru yang lebih tinggi, Anda mungkin menerima beberapa kesalahan kode status HTTP 503 (Layanan Tidak Tersedia).

Dengan namespace hierarkis, pembatas dalam kunci objek menjadi penting. Satu-satunya pembatas yang didukung adalah garis miring (`/`). Direktori ditentukan oleh batas pembatas. Misalnya, kunci

objek `dir1/dir2/file1.txt` menghasilkan direktori `dir1/` dan `dir2/` secara otomatis dibuat, dan objek `file1.txt` ditambahkan ke direktori `/dir2` di jalur `dir1/dir2/file1.txt`.

Direktori yang dibuat ketika objek diunggah ke bucket direktori tidak memiliki batas TPS per prefiks dan secara otomatis diskalakan sebelumnya untuk mengurangi kemungkinan kesalahan HTTP 503 (Layanan Tidak Tersedia). Penskalaan otomatis ini memungkinkan aplikasi Anda untuk memparalelkan permintaan baca dan tulis di dalam dan di seluruh direktori sesuai kebutuhan.

Paralelisasi permintaan penskalaan horizontal bucket direktori

Anda dapat mencapai kinerja terbaik dengan mengeluarkan beberapa permintaan sekaligus ke bucket direktori untuk menyebarkan permintaan Anda melalui koneksi terpisah untuk memaksimalkan bandwidth yang dapat diakses. S3 Express One Zone tidak memiliki batas untuk jumlah koneksi yang dilakukan ke bucket direktori Anda. Direktori individu dapat menskalakan kinerja secara horizontal dan otomatis ketika terjadi sejumlah besar penulisan bersamaan ke direktori yang sama.

Ketika kunci objek awalnya dibuat dan nama kuncinya termasuk direktori, direktori secara otomatis dibuat untuk objek tersebut. Unggahan objek berikutnya ke direktori yang sama tidak memerlukan direktori yang akan dibuat, yang mengurangi latensi pada unggahan objek ke direktori yang ada.

Meskipun struktur direktori dangkal dan dalam didukung untuk menyimpan objek dalam bucket direktori, bucket direktori secara otomatis menskalakan secara horizontal, dengan latensi yang lebih rendah pada unggahan bersamaan ke direktori yang sama atau ke saudara direktori paralel.

Gunakan autentikasi berbasis sesi

S3 Express One Zone dan bucket direktori mendukung mekanisme otorisasi berbasis sesi baru untuk mengautentikasi dan mengotorisasi permintaan ke bucket direktori. Dengan autentikasi berbasis sesi, SDK AWS secara otomatis menggunakan operasi API `CreateSession` untuk membuat token sesi sementara yang dapat digunakan untuk otorisasi latensi rendah permintaan data ke bucket direktori.

SDK AWS menggunakan operasi API `CreateSession` untuk meminta kredensial sementara, lalu secara otomatis membuat dan memuat ulang token untuk Anda atas nama Anda setiap 5 menit. Untuk memanfaatkan kelebihan kinerja kelas penyimpanan S3 Express One Zone, sebaiknya gunakan SDK AWS untuk memulai dan mengelola permintaan API `CreateSession`. Untuk informasi lebih lanjut tentang model berbasis sesi ini, lihat `CreateSession`otorisasi.

Praktik terbaik checksum tambahan S3

S3 Express One Zone menawarkan kepada Anda opsi untuk memilih algoritma checksum yang digunakan untuk memvalidasi data Anda selama mengunggah atau mengunduh. Anda dapat

memilih salah satu algoritma pemeriksaan integritas data Secure Hash Algorithms (SHA) atau Cyclic Redundancy Check (CRC) berikut: CRC32, CRC32C, SHA-1, dan SHA-256. Checksum berbasis MD5 tidak didukung dengan kelas penyimpanan S3 Express One Zone.

CRC32 adalah checksum default yang digunakan oleh SDK AWS saat mengirimkan data ke atau dari S3 Express One Zone. Kami merekomendasikan penggunaan CRC32 dan CRC32C untuk kinerja terbaik dengan kelas penyimpanan S3 Express One Zone.

Gunakan versi terbaru dari SDK AWS dan pustaka runtime umum

Beberapa SDK AWS juga menyediakan pustaka AWS Common Runtime (CRT) untuk lebih mempercepat kinerja di klien S3. SDK ini termasuk AWS SDK for Java 2.x, AWS SDK for C++, dan AWS SDK for Python (Boto3). Klien S3 berbasis CRT memindahkan objek ke dan dari S3 Express One Zone dengan kinerja dan keandalan yang ditingkatkan dengan secara otomatis menggunakan operasi API unggahan multibagian dan pengambilan rentang byte untuk mengotomatiskan koneksi penskalaan horizontal.

Untuk mencapai kinerja tertinggi dengan kelas penyimpanan S3 Express One Zone, kami menyarankan menggunakan versi terbaru SDK AWS yang menyertakan pustaka CRT atau menggunakan AWS Command Line Interface (AWS CLI).

Penyelesaian masalah kinerja

Permintaan coba lagi untuk aplikasi yang sensitif latensi

S3 Express One Zone dibuat khusus untuk menghadirkan tingkat kinerja tinggi yang konsisten tanpa penyetelan tambahan. Namun, menetapkan nilai batas waktu yang agresif dan percobaan ulang dapat membantu mendorong latensi dan kinerja yang konsisten. SDK AWS memiliki nilai batas waktu dan percobaan yang dapat dikonfigurasi yang dapat disesuaikan dengan toleransi aplikasi spesifik Anda.

Pustaka AWS Common Runtime (CRT) dan pemasangan jenis instans Amazon EC2

Aplikasi yang melakukan sejumlah besar operasi baca dan tulis cenderung membutuhkan lebih banyak memori atau kapasitas komputasi daripada aplikasi yang tidak melakukannya. Saat meluncurkan instans Amazon Elastic Compute Cloud (Amazon EC2) Anda untuk beban kerja yang menuntut kinerja, pilihlah jenis instans yang memiliki jumlah sumber daya yang dibutuhkan aplikasi Anda. Penyimpanan berkinerja tinggi S3 Express One Zone idealnya dipasangkan dengan jenis instans yang lebih besar dan lebih baru dengan jumlah memori sistem yang lebih besar serta CPU dan GPU yang lebih kuat yang dapat memanfaatkan penyimpanan berkinerja lebih tinggi. Kami juga

merekomendasikan penggunaan versi terbaru dari SDK AWS berkemampuan CRT, yang dapat mempercepat permintaan baca dan tulis secara paralel dengan lebih baik.

Gunakan autentikasi berbasis sesi di SDK AWS alih-alih HTTP API REST

Dengan Amazon S3, Anda juga dapat mengoptimalkan kinerja saat menggunakan permintaan HTTP API REST dengan mengikuti praktik terbaik yang sama yang merupakan bagian dari SDK AWS. Namun, dengan mekanisme otorisasi dan autentikasi berbasis sesi yang digunakan oleh S3 Express One Zone, kami sangat menyarankan Anda menggunakan SDK AWS untuk mengelola `CreateSession` dan token sesi yang dikelola. SDK AWS secara otomatis membuat dan memuat ulang token atas nama Anda menggunakan operasi API `CreateSession`. Menggunakan `CreateSession` simpanan pada latensi dua arah per permintaan ke AWS Identity and Access Management (IAM) untuk mengotorisasi setiap permintaan.

Mengembangkan dengan S3 Express One Zone

Amazon S3 Express One Zone adalah kelas penyimpanan S3 pertama di mana Anda dapat memilih satu Zona Ketersediaan dengan opsi untuk menempatkan penyimpanan objek bersama dengan sumber daya komputasi Anda, yang memberikan kecepatan akses setinggi mungkin. Dengan kelas penyimpanan S3 Express One Zone, Anda menggunakan bucket direktori S3 untuk menyimpan data. Setiap bucket direktori menggunakan kelas penyimpanan S3 Express One Zone untuk menyimpan objek dalam satu Zona Ketersediaan yang dapat Anda pilih saat membuat bucket.

Setelah membuat bucket direktori, Anda bisa langsung memulai pembacaan dan penulisan dengan latensi yang sangat rendah. Anda dapat berkomunikasi dengan bucket direktori menggunakan koneksi titik akhir melalui cloud privat virtual (VPC), atau menggunakan operasi API Zonal dan Regional untuk mengelola objek dan bucket direktori Anda. Anda juga dapat menggunakan kelas penyimpanan S3 Express One Zone melalui konsol Amazon S3, AWS Command Line Interface (AWS CLI), AWS SDK, dan dengan API REST Amazon S3.

Kelas penyimpanan Amazon S3 Express One Zone dirancang untuk ketersediaan 99,95 persen dalam satu Availability Zone dan didukung oleh Perjanjian Tingkat Layanan Amazon S3. Dengan S3 Express One Zone, data Anda disimpan secara berlebihan di beberapa perangkat dalam satu Zona Ketersediaan. S3 Express One Zone dirancang untuk menangani kegagalan perangkat bersamaan dengan cepat mendeteksi dan memperbaiki setiap redundansi yang hilang. Apabila perangkat yang ada mengalami kegagalan, S3 Express One Zone secara otomatis mengalihkan permintaan ke perangkat baru dalam Zona Ketersediaan. Redundansi ini membantu memastikan akses tanpa gangguan ke data Anda dalam Zona Ketersediaan.

Topik

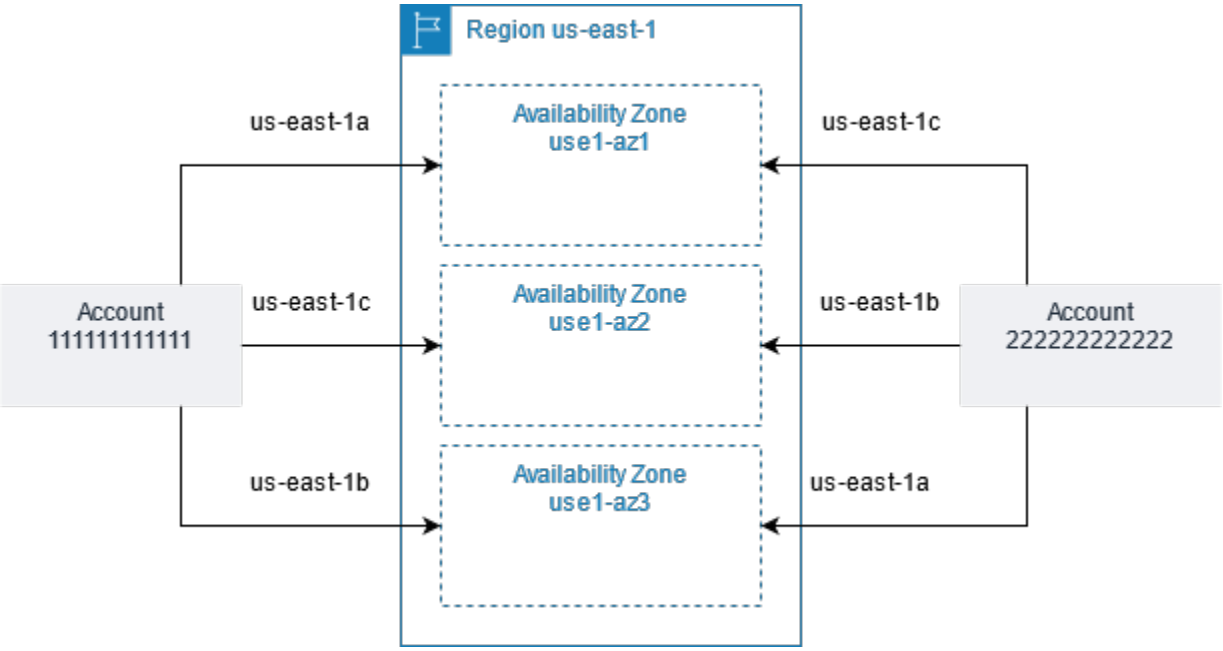
- Zona Ketersediaan dan Wilayah S3 Express One Zone
- Titik akhir Regional dan Zona
- Operasi S3 Express One Zone API

Zona Ketersediaan dan Wilayah S3 Express One Zone

Zona Ketersediaan adalah satu atau lebih pusat data diskret dengan daya redundan, jaringan, dan konektivitas dalam Wilayah AWS. Untuk mengoptimalkan pengambilan latensi rendah, objek di kelas penyimpanan Amazon S3 Express One Zone disimpan secara berlebihan di bucket direktori S3 dalam satu Zona Ketersediaan yang bersifat lokal untuk beban kerja komputasi Anda. Saat membuat bucket direktori, Anda memilih Availability Zone dan Wilayah AWS lokasi bucket Anda.

AWS memetakan Availability Zone fisik secara acak ke nama Availability Zone untuk masing-masing Akun AWS nama. Pendekatan ini membantu mendistribusikan sumber daya di seluruh Availability Zone di Wilayah AWS, alih-alih sumber daya yang kemungkinan terkonsentrasi di Availability Zone pertama untuk setiap Wilayah. Akibatnya, Availability Zone `us-east-1a` untuk Anda Akun AWS mungkin tidak mewakili lokasi fisik yang sama dengan lokasi `us-east-1a` yang berbeda Akun AWS. Untuk informasi selengkapnya, lihat Wilayah dan Zona Ketersediaan di Panduan Pengguna Amazon EC2.

Untuk mengoordinasikan Zona Ketersediaan di seluruh akun, Anda harus menggunakan ID AZ, yang merupakan pengenal unik dan konsisten untuk Zona Ketersediaan. Misalnya, `use1-az1` adalah ID AZ untuk `us-east-1` Wilayah dan memiliki lokasi fisik yang sama di setiap wilayah Akun AWS. Ilustrasi berikut menunjukkan kesamaan ID AZ (Zona Ketersediaan) untuk setiap akun, meskipun mungkin terdapat perbedaan dalam pemetaan nama Zona Ketersediaan untuk masing-masing akun.



Dengan S3 Express One Zone, data Anda disimpan secara berlebihan di beberapa perangkat dalam satu Zona Ketersediaan. S3 Express One Zone dirancang untuk ketersediaan 99,95 persen dalam satu Availability Zone dan didukung oleh Perjanjian Tingkat Layanan Amazon S3. Untuk informasi selengkapnya, lihat Zona Ketersediaan Tunggal

S3 Express One Zone didukung di Wilayah dan Zona Ketersediaan berikut:

Wilayah dan Zona Ketersediaan yang didukung S3 Express One Zone

Nama Wilayah	Kode Wilayah	ID Zona Ketersediaan
AS Timur (Virginia Utara)	`us-east-1`	`use1-az4`
		`use1-az5`
		`use1-az6`
AS Barat (Oregon)	`us-west-2`	`usw2-az1`
		`usw2-az3`
		`usw2-az4`

Nama Wilayah	Kode Wilayah	ID Zona Ketersediaan
Asia Pasifik (Tokyo)	`ap-northeast-1`	`apne1-az1`
		`apne1-az4`
Eropa (Stockholm)	`eu-north-1`	`eun1-az1`
		`eun1-az2`
		`eun1-az3`

Titik akhir Regional dan Zona

Untuk mengakses titik akhir Regional dan Zona Amazon S3 Express One Zone dari cloud privat virtual (VPC), Anda dapat menggunakan titik akhir VPC gateway. Setelah Anda membuat titik akhir gateway, Anda dapat menambahkannya sebagai target di tabel rute Anda untuk lalu lintas yang ditujukan dari VPC Anda ke S3 Express One Zone. Tidak ada biaya tambahan untuk menggunakan titik akhir gateway. Untuk informasi selengkapnya tentang cara mengonfigurasi titik akhir VPC gateway, lihat Jaringan untuk S3 Express One Zone.

Saat Anda bekerja dengan S3 Express One Zone, operasi API tingkat bucket (bidang kontrol) tersedia melalui titik akhir Regional dan disebut sebagai operasi API titik akhir Regional. Contoh operasi API titik akhir Regional adalah `CreateBucket` dan `DeleteBucket`.

Setelah membuat bucket direktori, Anda dapat menggunakan Zonal (level objek, atau operasi API titik akhir bidang data) untuk mengunggah dan mengelola objek di bucket direktori Anda. Operasi API titik akhir Zona tersedia melalui titik akhir Zona. Contoh operasi API Zona adalah `PutObject` dan `CopyObject`.

Operasi S3 Express One Zone API

Kelas penyimpanan Amazon S3 Express One Zone mendukung operasi API titik akhir Regional (tingkat bucket, atau bidang kontrol) dan Zonal (level objek, atau bidang data). Lihat informasi yang lebih lengkap di Jaringan untuk S3 Express One Zone dan Titik akhir dan titik akhir VPC gateway.

Operasi API titik akhir regional

Operasi API titik akhir Regional berikut didukung untuk S3 Express One Zone:

- CreateBucket
- DeleteBucket
- DeleteBucketPolicy
- GetBucketPolicy
- ListDirectoryBuckets
- PutBucketPolicy

Operasi API titik akhir zonal

Operasi API titik akhir Zonal berikut didukung untuk S3 Express One Zone:

- CreateSession
- CopyObject
- DeleteObject
- DeleteObjects
- GetObject
- GetObjectAttributes
- HeadBucket
- HeadObject
- ListObjectsV2
- PutObject
- AbortMultipartUpload
- CompleteMultiPartUpload
- CreateMultipartUpload
- ListMultipartUploads
- ListParts
- UploadPart
- UploadPartCopy

Mengelola akses data dengan titik akses Amazon S3

Jalur akses Amazon S3 menyederhanakan akses data untuk AWS layanan atau aplikasi pelanggan apa pun yang menyimpan data di S3. Titik akses diberi nama titik akhir jaringan yang dilampirkan ke bucket yang dapat Anda gunakan untuk melakukan operasi objek S3, seperti `GetObject` dan `PutObject`. Setiap titik akses memiliki izin dan kontrol jaringan berbeda yang diterapkan S3 untuk setiap permintaan yang dibuat melalui titik akses tersebut. Setiap titik akses memberlakukan kebijakan titik akses khusus yang bekerja bersama kebijakan bucket yang melekat pada bucket dasar. Anda dapat mengonfigurasi titik akses apa pun untuk menerima permintaan hanya dari cloud privat virtual (VPC) untuk membatasi akses data Amazon S3 ke jaringan pribadi. Anda juga dapat mengonfigurasi pengaturan blok akses publik khusus untuk setiap titik akses.

> **Note**
>
> - Anda hanya dapat menggunakan titik akses untuk melakukan pengoperasian pada objek. Anda tidak dapat menggunakan titik akses untuk melakukan operasi Amazon S3 lainnya, seperti mengubah atau menghapus bucket. Untuk mengetahui daftar lengkap pengoperasian S3 yang mendukung titik akses, lihat Kompatibilitas titik akses dengan AWS layanan.
> - Titik akses bekerja dengan beberapa, tetapi tidak semua, AWS layanan dan fitur. Misalnya, Anda tidak dapat mengonfigurasi Replikasi Lintas Wilayah untuk beroperasi melalui titik akses. Untuk daftar lengkap AWS layanan yang kompatibel dengan titik akses S3, lihatKompatibilitas titik akses dengan AWS layanan.

Bagian ini menjelaskan cara untuk bekerja dengan titik akses Amazon S3. Untuk informasi tentang bekerja dengan bucket, lihat Gambaran umum bucket. Untuk informasi tentang bekerja dengan objek, lihat Gambaran umum objek Amazon S3.

Topik

- Mengonfigurasi kebijakan IAM untuk menggunakan titik akses
- Membuat titik akses
- Menggunakan titik akses
- Pembatasan dan batasan titik akses

Mengonfigurasi kebijakan IAM untuk menggunakan titik akses

Kebijakan sumber daya dukungan jalur akses Amazon S3 AWS Identity and Access Management (IAM) yang memungkinkan Anda mengontrol penggunaan titik akses berdasarkan sumber daya, pengguna, atau kondisi lainnya. Agar aplikasi atau pengguna dapat mengakses objek melalui titik akses, baik titik akses maupun bucket yang mendasarinya harus mengizinkan permintaan tersebut.

> **Important**
>
> Menambahkan titik akses S3 ke bucket tidak mengubah perilaku bucket saat bucket diakses langsung melalui nama bucket atau Amazon Resource Name (ARN). Semua operasi yang ada terhadap bucket akan terus bekerja seperti sebelumnya. Pembatasan yang Anda sertakan dalam kebijakan titik akses hanya berlaku untuk permintaan yang dibuat melalui titik akses tersebut.

Saat Anda menggunakan kebijakan sumber daya IAM, pastikan untuk menyelesaikan peringatan keamanan, kesalahan, peringatan umum, dan saran AWS Identity and Access Management Access Analyzer sebelum Anda menyimpan kebijakan Anda. Penganalisis Akses IAM menjalankan pemeriksaan kebijakan untuk memvalidasi kebijakan Anda terhadap tata bahasa kebijakan IAM dan praktik terbaik. Pemeriksaan ini menghasilkan temuan dan memberikan rekomendasi untuk membantu Anda membuat kebijakan yang berfungsi dan sesuai dengan praktik terbaik keamanan.

Untuk mempelajari validasi kebijakan menggunakan Penganalisis Akses IAM lebih lanjut, lihat Memvalidasi kebijakan Penganalisis Akses IAM di Panduan Pengguna IAM. Untuk melihat daftar peringatan, kesalahan, dan saran yang ditampilkan oleh Penganalisis Akses IAM, lihat referensi pemeriksaan kebijakan Penganalisis Akses IAM.

Contoh kebijakan titik akses

Contoh berikut menunjukkan cara membuat kebijakan IAM untuk mengontrol permintaan yang dilakukan melalui titik akses.

> **Note**
>
> Izin yang diberikan dalam kebijakan titik akses hanya berlaku jika bucket yang mendasarinya juga mengizinkan akses yang sama. Anda dapat melakukan ini menggunakan dua cara:

1. (Disarankan) Delegasikan kontrol akses dari bucket ke titik akses seperti yang dijelaskan dalam Mendelegasikan kontrol akses ke titik akses.
2. Tambahkan izin yang sama yang tercantum dalam kebijakan titik akses ke kebijakan bucket dasar. Contoh kebijakan titik akses Contoh 1 mendemonstrasikan cara memodifikasi kebijakan bucket dasar agar akses yang diperlukan dapat dilakukan.

Example 1–Pemberian kebijakan titik akses

Kebijakan titik akses berikut memberikan izin kepada pengguna IAM *Jane* pada akun *123456789012* atas objek GET dan PUT dengan prefiks *Jane/* melalui titik akses *my-access-point* pada akun *123456789012*.

```
{
    "Version":"2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/Jane"
        },
        "Action": ["s3:GetObject", "s3:PutObject"],
        "Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/my-access-point/
object/Jane/*"
    }]
}
```

> **Note**
>
> Agar kebijakan titik akses secara efektif memberikan akses ke *Jane*, bucket yang mendasari juga harus mengizinkan akses yang sama ke *Jane*. Anda dapat mendelegasikan kontrol akses dari bucket ke titik akses seperti yang dijelaskan dalam Mendelegasikan kontrol akses ke titik akses. Atau, Anda dapat menambahkan kebijakan berikut ke bucket yang mendasarinya untuk memberikan izin yang diperlukan kepada Jane. Perhatikan bahwa entri `Resource` berbeda antara titik akses dan kebijakan bucket.
>
> ```
> {
>  "Version": "2012-10-17",
>  "Statement": [
> ```

```
    {
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/Jane"
        },
        "Action": ["s3:GetObject", "s3:PutObject"],
        "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/Jane/*"
    }]
}
```

Example 2–Kebijakan titik akses dengan kondisi tag

Kebijakan titik akses berikut memberikan izin *Mateo* kepada pengguna IAM dalam akun *123456789012* untuk objek GET melalui titik akses *my-access-point* dalam akun *123456789012* yang kunci tag *data* diatur dengan nilai *finance*.

```
{
    "Version":"2012-10-17",
    "Statement": [
    {
        "Effect":"Allow",
        "Principal" : {
            "AWS": "arn:aws:iam::123456789012:user/Mateo"
        },
        "Action":"s3:GetObject",
        "Resource" : "arn:aws:s3:us-west-2:123456789012:accesspoint/my-access-point/
object/*",
        "Condition" : {
            "StringEquals": {
                "s3:ExistingObjectTag/data": "finance"
            }
        }
    }]
}
```

Example 3–Kebijakan titik akses yang mengizinkan daftar bucket

Kebijakan titik akses berikut memungkinkan pengguna IAM `Arnav` di akun *123456789012* untuk melihat objek yang ada dalam bucket yang mendasari titik akses *my-access-point* pada akun *123456789012*.

```
{
    "Version":"2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/Arnav"
        },
        "Action": "s3:ListBucket",
        "Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/my-access-point"
    }]
}
```

Example 4–Kebijakan kontrol layanan

Kebijakan kontrol layanan berikut mewajibkan semua titik akses baru untuk dibuat dengan asal jaringan cloud privat virtual (VPC). Dengan kebijakan ini, pengguna di dalam organisasi Anda tidak dapat membuat titik akses baru yang dapat diakses dari internet.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Deny",
        "Action": "s3:CreateAccessPoint",
        "Resource": "*",
        "Condition": {
            "StringNotEquals": {
                "s3:AccessPointNetworkOrigin": "VPC"
            }
        }
    }]
}
```

Example 5–Kebijakan bucket untuk membatasi operasi S3 ke asal jaringan VPC

Kebijakan bucket berikut membatasi akses ke semua operasi S3 object untuk bucket *example-s3-bucket* ke titik akses dengan asal jaringan VPC.

> **Important**
>
> Sebelum menggunakan pernyataan seperti yang ditunjukkan pada contoh ini, pastikan Anda tidak perlu menggunakan fitur yang tidak didukung oleh titik akses, seperti Cross-Region Replication.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:BypassGovernanceRetention",
                "s3:DeleteObject",
                "s3:DeleteObjectTagging",
                "s3:DeleteObjectVersion",
                "s3:DeleteObjectVersionTagging",
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:GetObjectLegalHold",
                "s3:GetObjectRetention",
                "s3:GetObjectTagging",
                "s3:GetObjectVersion",
                "s3:GetObjectVersionAcl",
                "s3:GetObjectVersionTagging",
                "s3:ListMultipartUploadParts",
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:PutObjectLegalHold",
                "s3:PutObjectRetention",
                "s3:PutObjectTagging",
                "s3:PutObjectVersionAcl",
                "s3:PutObjectVersionTagging",
                "s3:RestoreObject"
            ],
            "Resource": "arn:aws:s3:::example-s3-bucket/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:AccessPointNetworkOrigin": "VPC"
```

```
                }
            }
        }
    ]
}
```

Kunci syarat

Titik akses S3 memiliki kunci kondisi yang dapat digunakan dalam kebijakan IAM untuk mengontrol akses ke sumber daya Anda. Kunci kondisi berikut hanya mewakili bagian dari kebijakan IAM. Untuk contoh kebijakan lengkap, lihat Contoh kebijakan titik akses, the section called “Mendelegasikan kontrol akses ke titik akses”, dan the section called “Memberikan izin untuk titik akses lintas akun”.

`s3:DataAccessPointArn`

Contoh ini menunjukkan string yang dapat Anda gunakan untuk mencocokkan ARN titik akses. Contoh berikut cocok dengan semua titik akses untuk Akun AWS *123456789012*di Wilayah *us-west-2*:

```
"Condition" : {
    "StringLike": {
        "s3:DataAccessPointArn": "arn:aws:s3:us-west-2:123456789012:accesspoint/*"
    }
}
```

`s3:DataAccessPointAccount`

Contoh ini menunjukkan operator string yang dapat Anda gunakan untuk mencocokkan ID akun pemilik titik akses. Contoh berikut cocok dengan semua titik akses yang dimiliki oleh Akun AWS *123456789012*.

```
"Condition" : {
    "StringEquals": {
        "s3:DataAccessPointAccount": "123456789012"
    }
}
```

`s3:AccessPointNetworkOrigin`

Contoh ini menunjukkan operator string yang dapat Anda gunakan untuk mencocokkan asal jaringan, baik `Internet` atau `VPC`. Contoh berikut hanya mencocokkan titik akses dengan asal VPC.

```
"Condition" : {
    "StringEquals": {
        "s3:AccessPointNetworkOrigin": "VPC"
    }
}
```

Untuk informasi selengkapnya tentang menggunakan kunci kondisi dengan Amazon S3, lihat Tindakan, sumber daya, dan kunci kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Mendelegasikan kontrol akses ke titik akses

Anda dapat mendelegasikan kontrol akses untuk sebuah bucket ke titik akses bucket. Kebijakan ini memungkinkan akses penuh ke semua titik akses yang dimiliki oleh akun pemilik bucket. Oleh karena itu, semua akses ke bucket ini dikendalikan oleh kebijakan yang melekat pada titik aksesnya. Kami menyarankan Anda untuk mengonfigurasi bucket dengan cara ini untuk semua kasus penggunaan yang tidak memerlukan akses langsung ke bucket.

Example 6–Kebijakan bucket yang mendelegasikan kontrol akses ke titik akses

```
{
    "Version": "2012-10-17",
    "Statement" : [
    {
        "Effect": "Allow",
        "Principal" : { "AWS": "*" },
        "Action" : "*",
        "Resource" : [ "Bucket ARN", "Bucket ARN/*"],
        "Condition": {
            "StringEquals" : { "s3:DataAccessPointAccount" : "Bucket owner's account
 ID" }
        }
    }]
}
```

Memberikan izin untuk titik akses lintas akun

Untuk membuat titik akses ke bucket yang dimiliki oleh akun lain, Anda harus terlebih dahulu membuat titik akses dengan menentukan nama bucket dan ID pemilik akun. Kemudian, pemilik bucket harus memperbarui kebijakan bucket untuk mengotorisasi permintaan dari titik akses. Membuat titik akses mirip dengan membuat DNS CNAME di mana titik akses tidak menyediakan akses ke konten bucket. Semua akses bucket dikendalikan oleh kebijakan bucket. Contoh kebijakan bucket berikut mengizinkan `GET` dan `LIST` meminta pada bucket dari titik akses yang dimiliki oleh Akun AWS yang tepercaya.

Ganti *`Bucket ARN`* dengan ARN ember.

Example 7 — Kebijakan bucket mendelegasikan izin ke yang lain Akun AWS

```
{
    "Version": "2012-10-17",
    "Statement" : [
    {
        "Effect": "Allow",
        "Principal" : { "AWS": "*" },
        "Action" : ["s3:GetObject","s3:ListBucket"],
        "Resource" : [ "Bucket ARN", "Bucket ARN/*"],
        "Condition": {
            "StringEquals" : { "s3:DataAccessPointAccount" : "Access point owner's
 account ID" }
        }
    }]
}
```

Membuat titik akses

Amazon S3 menyediakan fungsionalitas untuk membuat, serta mengelola titik akses. Anda dapat membuat titik akses S3 dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), AWS SDK, atau Amazon S3 REST API.

Secara default, Anda dapat membuat hingga 10.000 titik akses per Wilayah untuk setiap Akun AWS Anda. Jika Anda memerlukan lebih dari 10.000 titik akses untuk satu akun di satu Wilayah, Anda dapat meminta peningkatan kuota layanan. Untuk informasi lebih lanjut tentang kuota layanan dan permintaan peningkatan, lihat AWS Service Quotas dalam Referensi Umum AWS.

> **Note**
>
> Karena Anda mungkin ingin mempublikasikan nama titik akses Anda sehingga pengguna lain dapat menggunakan titik akses tersebut, hindari memasukkan informasi sensitif dalam nama titik akses. Nama titik akses diterbitkan dalam basis data yang dapat diakses publik yang dikenal sebagai Sistem Nama Domain (DNS).

Aturan penamaan titik akses Amazon S3

Nama titik akses harus memenuhi ketentuan berikut:

- Harus unik dalam satu Akun AWS dan Wilayah
- Harus mematuhi pembatasan penamaan DNS
- Harus dimulai dengan angka atau huruf kecil
- Panjang nama harus berkisar antara 3 sampai 50 karakter
- Tidak dapat diawali atau diakhiri dengan tanda hubung (`-`)
- Tidak boleh berisi garis bawah (_), huruf besar, atau titik (`.`)
- Tidak boleh diakhiri dengan sufiks `-s3alias`. Sufiks ini dicadangkan untuk nama alias titik akses. Untuk informasi selengkapnya, lihat Menggunakan alias gaya bucket untuk titik akses bucket S3 Anda.

Untuk membuat titik akses, lihat topik berikut ini.

Topik

- Membuat titik akses
- Membuat titik akses terbatas pada cloud privat virtual
- Mengelola akses publik ke titik akses

Membuat titik akses

Titik akses terkait dengan persis satu bucket Amazon S3. Jika Anda ingin menggunakan ember di dalam Anda Akun AWS, Anda harus terlebih dahulu membuat ember. Untuk informasi tentang membuat bucket, lihat Membuat, mengonfigurasi, dan bekerja dengan bucket Amazon S3.

Anda juga dapat membuat titik akses lintas akun yang terkait dengan bucket di bucket lain Akun AWS, selama Anda mengetahui nama bucket dan ID akun pemilik bucket. Namun, membuat titik akses lintas akun tidak memberikan Anda akses ke data di bucket sampai Anda diberikan izin dari pemilik bucket. Pemilik bucket harus memberikan akses akun pemilik titik akses (akun Anda) ke bucket melalui kebijakan bucket. Untuk informasi selengkapnya, lihat Memberikan izin untuk titik akses lintas akun.

Secara default, Anda dapat membuat hingga 10.000 titik akses per Wilayah untuk setiap Akun AWS Anda. Jika Anda memerlukan lebih dari 10.000 titik akses untuk satu akun di satu Wilayah, Anda dapat meminta peningkatan kuota layanan. Untuk informasi lebih lanjut tentang kuota layanan dan permintaan peningkatan, lihat AWS Service Quotas dalam Referensi Umum AWS.

Contoh berikut menunjukkan cara membuat titik akses dengan AWS CLI dan konsol S3. Untuk informasi lebih lanjut tentang cara membuat titik akses menggunakan API REST, lihat CreateAccessPoint dalam Referensi Amazon Simple Storage Service API.

Menggunakan konsol S3

Untuk membuat titik akses

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di bilah navigasi di bagian atas halaman, pilih nama yang saat ini ditampilkan Wilayah AWS. Selanjutnya, pilih Wilayah tempat Anda ingin membuat titik akses.
3. Pada panel navigasi di kiri, pilih Titik Akses.
4. Pada halaman Titik Akses, pilih Buat titik akses.
5. Di dalam bidang Nama titik akses, masukkan nama untuk titik akses tersebut. Untuk informasi lebih lanjut tentang penamaan titik akses, lihat Aturan penamaan titik akses Amazon S3.
6. Untuk Nama bucket, tentukan bucket S3 yang ingin Anda gunakan dengan titik akses.

   Untuk menggunakan bucket di dalam akun Anda, pilih Pilih bucket di akun ini, dan masukkan atau telusuri nama bucket.

   Untuk menggunakan bucket di tempat lain Akun AWS, pilih Tentukan bucket di akun lain, lalu masukkan Akun AWS ID dan nama bucket.

> **Note**
>
> Jika Anda menggunakan bucket di bucket lain Akun AWS, pemilik bucket harus memperbarui kebijakan bucket untuk mengotorisasi permintaan dari titik akses. Untuk contoh kebijakan bucket, lihat Memberikan izin untuk titik akses lintas akun.

7. Pilih Asal jaringan. Jika Anda memilih Cloud privat virtual (VPC), masukkan ID VPC yang ingin Anda gunakan dengan titik akses.

   Untuk informasi lebih lanjut tentang asal jaringan untuk titik akses, lihat Membuat titik akses terbatas pada cloud privat virtual.

8. Di bagian bawah Pengaturan Blokir Titik Akses untuk Titik Akses ini, pilih pengaturan blokir akses publik yang ingin Anda terapkan ke titik akses tersebut. Semua pengaturan blok akses publik diaktifkan secara default untuk titik akses baru. Kami menyarankan Anda tetap mengaktifkan semua pengaturan kecuali Anda tahu bahwa Anda memiliki kebutuhan khusus untuk menonaktifkan salah satu pengaturan tersebut.

   > **Note**
   >
   > Setelah Anda membuat titik akses, Anda tidak dapat mengubah pengaturan blokir akses publik.

   Untuk informasi selengkapnya tentang menggunakan Blokir Akses Publik Amazon S3 dengan titik akses, lihat Mengelola akses publik ke titik akses.

9. (Opsional) Di bawah Kebijakan Titik Akses-opsional, tentukan kebijakan titik akses. Sebelum Anda menyimpan kebijakan Anda, pastikan untuk mengatasi peringatan keamanan, kesalahan, peringatan umum, dan saran apa pun. Untuk informasi selengkapnya tentang penentuan kebijakan titik akses, lihat Contoh kebijakan titik akses.
10. Pilih Buat titik akses.

Menggunakan AWS CLI

Contoh berikut membuat Titik Akses Lambda Objek yang diberi nama *example-ap* untuk bucket *DOC-EXAMPLE-BUCKET* di akun *111122223333*. Untuk membuat titik akses, Anda mengirim permintaan ke Amazon S3 yang menentukan hal berikut:

- Nama titik akses. Untuk informasi tentang aturan penamaan, lihat the section called "Aturan penamaan titik akses Amazon S3".
- Nama bucket yang ingin Anda asosiasikan dengan titik akses.
- ID akun untuk Akun AWS yang memiliki ember.

```
aws s3control create-access-point --name example-ap --account-id 111122223333 --
bucket DOC-EXAMPLE-BUCKET
```

Saat Anda membuat titik akses dengan menggunakan bucket di tempat lain Akun AWS, sertakan `--bucket-account-id` parameternya. Contoh perintah berikut membuat titik akses di Akun AWS *111122223333*, menggunakan bucket *DOC-EXAMPLE-BUCKET2*, yang ada di Akun AWS *444455556666*.

```
aws s3control create-access-point --name example-ap --account-id 111122223333 --
bucket DOC-EXAMPLE-BUCKET --bucket-account-id 444455556666
```

Membuat titik akses terbatas pada cloud privat virtual

Saat membuat titik akses, Anda dapat memilih untuk membuat titik akses dapat diakses dari internet, atau Anda dapat menentukan bahwa semua permintaan yang dibuat melalui titik akses tersebut harus berasal dari cloud privat virtual (VPC) tertentu. Titik akses yang dapat diakses dari internet dikatakan memiliki asal jaringan dari `Internet`. Ini dapat digunakan dari mana saja di internet, dengan tunduk pada pembatasan akses lainnya yang berlaku untuk titik akses, bucket yang mendasarinya, dan sumber daya terkait, seperti objek yang diminta. Titik akses yang hanya dapat diakses dari VPC yang ditentukan memiliki asal jaringan `VPC`, dan Amazon S3 akan menolak permintaan apa pun yang diajukan ke titik akses yang tidak berasal dari VPC tersebut.

> **Important**
>
> Anda hanya dapat menentukan asal jaringan titik akses saat membuat titik akses. Setelah membuat titik akses, Anda tidak dapat mengubah asal jaringannya.

Untuk membatasi titik akses ke akses kustom VPC, Anda harus menyertakan parameter `VpcConfiguration` dengan permintaan untuk membuat titik akses. Dalam parameter `VpcConfiguration`, Anda menentukan ID VPC yang ingin Anda bisa gunakan titik aksesnya. Jika

permintaan dibuat melalui titik akses, permintaan harus berasal dari VPC atau Amazon S3 akan menolaknya.

Anda dapat mengambil asal jaringan titik akses menggunakan AWS CLI, AWS SDK, atau REST API. Jika titik akses memiliki konfigurasi VPC yang ditentukan, asal jaringannya adalah `VPC`. Jika tidak, asal jaringan titik aksesnya adalah `Internet`.

Example

Contoh: Membuat titik akses yang dibatasi untuk akses VPC

Contoh berikut membuat titik akses yang bernama `example-vpc-ap` untuk bucket `example-bucket` dalam akun `123456789012` yang mengizinkan akses hanya dari VPC `vpc-1a2b3c`. Contoh tersebut kemudian memverifikasi bahwa titik akses yang baru memiliki asal jaringan `VPC`.

AWS CLI

```
aws s3control create-access-point --name example-vpc-ap --account-id 123456789012 --
bucket example-bucket --vpc-configuration VpcId=vpc-1a2b3c
```

```
aws s3control get-access-point --name example-vpc-ap --account-id 123456789012

{
    "Name": "example-vpc-ap",
    "Bucket": "example-bucket",
    "NetworkOrigin": "VPC",
    "VpcConfiguration": {
        "VpcId": "vpc-1a2b3c"
    },
    "PublicAccessBlockConfiguration": {
        "BlockPublicAcls": true,
        "IgnorePublicAcls": true,
        "BlockPublicPolicy": true,
        "RestrictPublicBuckets": true
    },
    "CreationDate": "2019-11-27T00:00:00Z"
}
```

Untuk menggunakan titik akses dengan VPC, Anda harus mengubah kebijakan akses untuk titik akhir VPC Anda. Titik akhir VPC memungkinkan lalu lintas mengalir dari VPC Anda ke Amazon S3. Kebijakan kontrol akses yang mengontrol bagaimana sumber daya dalam VPC diizinkan untuk

berinteraksi dengan Amazon S3. Permintaan dari VPC ke Amazon S3 hanya berhasil melalui titik akses jika kebijakan titik akhir VPC memberikan akses ke titik akses dan bucket yang mendasarinya.

> **Note**
>
> Agar sumber daya hanya dapat diakses dalam VPC, pastikan untuk membuat zona yang di-hosting pribadi untuk titik akhir VPC Anda. Untuk menggunakan zona yang di-hosting pribadi, modifikasi pengaturan VPC Anda sehingga atribut jaringan VPC `enableDnsHostnames` dan `enableDnsSupport` diatur ke `true`.

Contoh pernyataan kebijakan berikut ini mengonfigurasi titik akhir VPC untuk mengizinkan panggilan ke `GetObject` untuk bucket bernama `awsexamplebucket1` dan titik akses bernama `example-vpc-ap`.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Principal": "*",
        "Action": [
            "s3:GetObject"
        ],
        "Effect": "Allow",
        "Resource": [
            "arn:aws:s3:::awsexamplebucket1/*",
            "arn:aws:s3:us-west-2:123456789012:accesspoint/example-vpc-ap/object/*"
        ]
    }]
}
```

> **Note**
>
> Pernyataan `"Resource"` di dalam contoh ini menggunakan Amazon Resource Name (ARN) untuk menentukan titik akses. Untuk informasi lebih lanjut tentang titik akses ARN, lihat Menggunakan titik akses.

Untuk informasi selengkapnya tentang kebijakan titik akhir VPC, lihat Menggunakan kebijakan titik akhir untuk Amazon S3 dalam Panduan Pengguna Amazon VPC.

Mengelola akses publik ke titik akses

Titik akses Amazon S3 mendukung pengaturan independen blokir akses publik untuk setiap titik akses. Saat membuat titik akses, Anda dapat menentukan pengaturan blokir akses publik yang berlaku pada titik akses tersebut. Untuk setiap permintaan yang dibuat melalui titik akses, Amazon S3 mengevaluasi pengaturan blokir akses publik untuk titik akses tersebut, bucket yang mendasarinya, dan akun pemilik bucket. Jika salah satu pengaturan ini menunjukkan bahwa permintaan tersebut harus diblokir, Amazon S3 menolak permintaan tersebut.

Untuk informasi lebih lanjut tentang fitur Blokir Akses Publik S3, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

> **Important**
>
> - Semua pengaturan blokir akses publik diaktifkan secara default untuk titik akses baru. Anda harus secara eksplisit menonaktifkan pengaturan apa pun yang tidak ingin Anda terapkan ke titik akses.
> - Saat ini, Amazon S3 tidak mendukung perubahan pengaturan akses publik blokir titik akses setelah titik akses dibuat.

Example

Contoh: Membuat titik akses dengan Pengaturan Blok Akses Publik Khusus

Contoh ini membuat titik akses bernama `example-ap` untuk bucket `example-bucket` dalam akun `123456789012` dengan pengaturan Blokir Akses Publik non-default. Contoh ini kemudian mengambil konfigurasi titik akses baru untuk memverifikasi pengaturan Blokir Akses Publik miliknya.

AWS CLI

```
aws s3control create-access-point --name example-ap --account-id
 123456789012 --bucket example-bucket --public-access-block-configuration
 BlockPublicAcls=false,IgnorePublicAcls=false,BlockPublicPolicy=true,RestrictPublicBuckets=t
```

```
aws s3control get-access-point --name example-ap --account-id 123456789012

{
    "Name": "example-ap",
```

```
    "Bucket": "example-bucket",
    "NetworkOrigin": "Internet",
    "PublicAccessBlockConfiguration": {
        "BlockPublicAcls": false,
        "IgnorePublicAcls": false,
        "BlockPublicPolicy": true,
        "RestrictPublicBuckets": true
    },
    "CreationDate": "2019-11-27T00:00:00Z"
}
```

Menggunakan titik akses

Anda dapat mengakses objek di bucket Amazon S3 dengan titik akses menggunakan,, AWS SDK AWS Management Console AWS CLI, atau S3 REST API.

Titik akses memiliki Amazon Resource Names (ARN). Titik akses ARN serupa dengan bucket ARN, tetapi diketik secara eksplisit dan menyandikan Wilayah titik akses dan ID Akun AWS pemilik titik akses. Untuk informasi selengkapnya tentang ARN, lihat Amazon Resource Name (ARN) di Referensi Umum AWS.

Titik akses ARN menggunakan format `arn:aws:s3:`*`region`*`:`*`account-id`*`:accesspoint/`*`resource`*. Sebagai contoh:

- arn:aws:s3:us-west-2:123456789012:accesspoint/test mewakili titik akses bernama `test`, yang dimiliki oleh akun `123456789012` di Wilayah `us-west-2`.
- arn:aws:s3:us-west-2:123456789012:accesspoint/* mewakili semua titik akses dalam akun `123456789012` di Wilayah `us-west-2`.

ARN untuk objek yang diakses melalui titik akses menggunakan format `arn:aws:s3:`*`region`*`:`*`account-id`*`:accesspoint/`*`access-point-name`*`/object/`*`resource`*. Sebagai contoh:

- arn:aws:s3:us-west-2:123456789012:accesspoint/test/object/unit-01 mewakili objek `unit-01`, yang diakses melalui titik akses bernama `test`, yang dimiliki oleh akun `123456789012` di Wilayah `us-west-2`.
- arn:aws:s3:us-west-2:123456789012:accesspoint/test/object/* mewakili semua objek untuk titik akses `test`, pada akun `123456789012` di Wilayah `us-west-2`.

- arn:aws:s3:us-west-2:123456789012:accesspoint/test/object/unit-01/finance/* mewakili semua objek di bawah prefiks `unit-01/finance/` untuk titik akses `test`, dalam akun `123456789012` di Wilayah `us-west-2`.

Mengakses bucket melalui titik akses S3

Jalur akses S3 hanya mendukung virtual-host-style pengalamatan. Untuk mengatur bucket melalui sebuah titik akses, gunakan format berikut.

```
https://AccessPointName-AccountId.s3-accesspoint.region.amazonaws.com
```

> **Note**
>
> - Jika nama titik akses Anda mencakup karakter tanda pisah (-), sertakan tanda pisah dalam URL dan masukkan tanda pisah lainnya sebelum ID akun. Misalnya, untuk menggunakan titik akses bernama `finance-docs` yang dimiliki oleh akun `123456789012` di Wilayah `us-west-2`, URL yang sesuai yaitu `https://finance-docs-123456789012.s3-accesspoint.us-west-2.amazonaws.com`.
> - Titik akses S3 tidak mendukung akses oleh HTTP, hanya akses aman oleh HTTPS.

Topik

- Pemantauan dan pencatatan titik akses
- Menggunakan titik akses Amazon S3 dengan konsol Amazon S3
- Menggunakan alias gaya bucket untuk titik akses bucket S3 Anda
- Menggunakan titik akses dengan operasi Amazon S3 yang kompatibel

Jika Anda memiliki Cloud Privat Virtual (VPC), lihat Mengelola akses Amazon S3 dengan titik akhir VPC dan Titik Akses S3.

Pemantauan dan pencatatan titik akses

Permintaan log Amazon S3 yang dibuat melalui titik akses dan permintaan yang dibuat ke API yang mengelola titik akses, seperti `CreateAccessPoint` dan `GetAccessPointPolicy`. Untuk

memantau dan mengelola pola penggunaan, Anda juga dapat mengonfigurasi metrik permintaan CloudWatch Log Amazon untuk titik akses.

Topik

- CloudWatch metrik permintaan
- Log permintaan

CloudWatch metrik permintaan

Untuk memahami dan meningkatkan kinerja aplikasi yang menggunakan titik akses, Anda dapat menggunakan CloudWatch metrik permintaan Amazon S3. Metrik permintaan membantu Anda memantau permintaan Amazon S3 untuk mengidentifikasi dan bertindak terhadap masalah operasional dengan cepat.

Secara default, metrik permintaan tersedia di tingkat bucket. Namun, Anda dapat menentukan filter untuk metrik permintaan menggunakan prefiks bersama, tag objek, atau titik akses. Saat Anda membuat filter titik akses, konfigurasi metrik permintaan menyertakan permintaan ke titik akses yang Anda tentukan. Anda dapat menerima metrik, mengatur alarm, dan mengakses dasbor untuk melihat operasi waktu nyata yang dilakukan melalui titik akses ini.

Anda harus memilih untuk meminta metrik dengan mengonfigurasinya di dalam konsol, atau menggunakan API Amazon S3. Metrik permintaan tersedia dalam interval 1 menit setelah beberapa latensi untuk diproses. Metrik permintaan ditagih dengan tarif yang sama dengan metrik CloudWatch kustom. Untuk informasi lebih lanjut, lihat harga Amazon CloudWatch .

Untuk membuat konfigurasi metrik permintaan yang memfilter berdasarkan titik akses, lihat Membuat konfigurasi metrik yang melakukan filter berdasarkan prefiks, tag objek, atau titik akses.

Log permintaan

Anda dapat mencatat permintaan yang dibuat melalui titik akses dan permintaan yang dibuat ke API yang mengelola titik akses, seperti `CreateAccessPoint` dan `GetAccessPointPolicy`, dengan menggunakan pencatatan akses server dan AWS CloudTrail.

CloudTrail entri log untuk permintaan yang dibuat melalui titik akses termasuk titik akses ARN di `resources` bagian log.

Sebagai contoh, anggap Anda memiliki konfigurasi berikut:

- Sebuah bucket bernama `DOC-EXAMPLE-BUCKET1` di Wilayah `us-west-2` yang berisi objek `my-image.jpg`
- Titik akses bernama `my-bucket-ap` yang terkait dengan `DOC-EXAMPLE-BUCKET1`
- Akun AWS ID dari `123456789012`

Contoh berikut menunjukkan `resources` bagian dari entri CloudTrail log untuk konfigurasi sebelumnya:

```
"resources": [
        {"type": "AWS::S3::Object",
            "ARN": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/my-image.jpg"
        },
        {"accountId": "123456789012",
            "type": "AWS::S3::Bucket",
            "ARN": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
        },
        {"accountId": "123456789012",
            "type": "AWS::S3::AccessPoint",
            "ARN": "arn:aws:s3:us-west-2:123456789012:accesspoint/my-bucket-ap"
        }
    ]
```

Untuk informasi selengkapnya tentang Log Akses Server S3, lihat Pencatatan permintaan dengan pencatatan akses server. Untuk informasi lebih lanjut tentang AWS CloudTrail, lihat Apa itu AWS CloudTrail? dalam AWS CloudTrail User Guide.

Menggunakan titik akses Amazon S3 dengan konsol Amazon S3

Bagian ini menjelaskan cara mengelola dan menggunakan titik akses Amazon S3 Anda menggunakan AWS Management Console. Sebelum memulai, buka halaman detail untuk titik akses yang ingin Anda kelola atau gunakan, seperti yang dijelaskan dalam prosedur berikut.

Topik

- Mencantumkan titik akses untuk akun Anda
- Membuat daftar titik akses untuk bucket
- Melihat detail konfigurasi untuk titik akses
- Menggunakan titik akses
- Melihat pengaturan blokir akses publik untuk titik akses

- Mengedit kebijakan titik akses
- Menghapus titik akses

Mencantumkan titik akses untuk akun Anda

Untuk mencantumkan semua titik akses yang dibuat di Akun AWS

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di bilah navigasi di bagian atas halaman, pilih nama yang saat ini ditampilkan Wilayah AWS. Selanjutnya, pilih Wilayah yang ingin Anda daftarkan titik akses.
3. Di panel navigasi di sisi kiri konsol, pilih titik akses.
4. Pada halaman titik akses, di bawah titik akses, lihat titik akses di Anda Wilayah AWS.
5. (Opsional) Mencari titik akses berdasarkan nama, dengan memasukkan nama ke bidang teks di samping menu pilihan menurun Wilayah.
6. Pilih nama titik akses yang ingin Anda kelola atau gunakan.

Membuat daftar titik akses untuk bucket

Untuk mencantumkan semua titik akses di dalam Anda Akun AWS untuk satu ember

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di bilah navigasi di bagian atas halaman, pilih nama yang saat ini ditampilkan Wilayah AWS lalu pilih Wilayah yang ingin Anda daftarkan titik akses.
3. Di panel navigasi di sisi kiri konsol, pilih Bucket.
4. Di halaman Bucket, pilih nama bucket yang titik aksesnya ingin Anda cantumkan.
5. Pada halaman detail bucket, pilih tab titik akses.
6. Pilih nama titik akses yang ingin Anda kelola atau gunakan.

Melihat detail konfigurasi untuk titik akses

1. Navigasi ke halaman detail titik akses untuk titik akses yang detailnya ingin Anda lihat, seperti yang dijelaskan di Mencantumkan titik akses untuk akun Anda.

2. Di bagian bawah gambaran umum titik akses, lihat detail konfigurasi dan properti untuk titik akses yang dipilih.

Menggunakan titik akses

1. Navigasi ke halaman detail titik akses untuk titik akses yang detailnya ingin Anda lihat, seperti yang dijelaskan di Mencantumkan titik akses untuk akun Anda.
2. Di bawah tab Objek, pilih nama objek yang ingin Anda akses melalui titik akses. Pada halaman operasi objek, konsol menampilkan label di atas nama bucket Anda yang menunjukkan titik akses yang saat ini Anda gunakan. Saat Anda menggunakan titik akses, Anda hanya dapat melakukan operasi objek yang diizinkan oleh izin titik akses.

> **Note**
>
> - Tampilan konsol selalu menampilkan semua objek dalam bucket. Menggunakan titik akses seperti yang dijelaskan dalam prosedur ini membatasi operasi yang dapat Anda lakukan pada objek tersebut, namun tidak membatasi apakah Anda dapat melihat keberadaan objek tersebut di dalam bucket.
> - S3 Management Console tidak mendukung penggunaan titik akses cloud privat virtual (VPC) untuk mengakses sumber daya bucket. Untuk mengakses sumber daya bucket dari titik akses VPC, gunakan API REST AWS CLI, AWS SDK, atau Amazon S3.

Melihat pengaturan blokir akses publik untuk titik akses

1. Navigasi ke halaman detail titik akses untuk titik akses yang detailnya ingin Anda lihat, seperti yang dijelaskan di Mencantumkan titik akses untuk akun Anda.
2. Pilih Izin.
3. Di bagian bawah kebijakan titik akses, tinjau pengaturan Blokir Akses Publik.

> **Note**
>
> Anda tidak dapat mengubah pengaturan Blokir Akses Publik untuk titik akses setelah titik akses dibuat.

Mengedit kebijakan titik akses

1. Navigasi ke halaman detail titik akses untuk titik akses yang detailnya ingin Anda lihat, seperti yang dijelaskan di Mencantumkan titik akses untuk akun Anda.
2. Pilih Izin.
3. Di bagian bawah kebijakan titik akses, pilih Edit.
4. Masukkan kebijakan titik akses di kolom teks. Konsol secara otomatis menampilkan Amazon Resource Name (ARN) untuk titik akses, yang dapat Anda gunakan dalam kebijakan.

Menghapus titik akses

1. Navigasikan ke daftar titik akses untuk akun Anda atau untuk bucket tertentu, seperti yang dijelaskan dalam Mencantumkan titik akses untuk akun Anda.
2. Pilih tombol opsi di sebelah nama titik akses yang ingin Anda hapus.
3. Pilih Hapus.
4. Konfirmasikan bahwa Anda ingin menghapus titik akses Anda dengan memasukkan namanya di bidang teks yang muncul, dan pilih Hapus.

Menggunakan alias gaya bucket untuk titik akses bucket S3 Anda

Saat Anda membuat titik akses, Amazon S3 secara otomatis membuat alias yang dapat Anda gunakan alih-alih nama bucket Amazon S3 untuk akses data. Anda dapat menggunakan alias titik akses ini alih-alih Amazon Resource Name (ARN) untuk operasi bidang data titik akses. Untuk mengetahui daftar operasi ini, lihat Kompatibilitas titik akses dengan AWS layanan.

Contoh berikut menunjukkan ARN dan alias titik akses untuk titik akses bernama *my-access-point*.

- ARN–`arn:aws:s3:`*`region`*`:`*`account-id`*`:accesspoint/`*`my-access-point`*
- Alias titik akses–*`my-access-point`*`-`*`hrzrlukc5m36ft7okagglf3gmwluquse1b`*`-s3alias`

Untuk informasi selengkapnya tentang ARN, lihat Amazon Resource Name (ARN) di Referensi Umum AWS.

Nama alias titik akses

Nama alias titik akses dibuat dalam namespace yang sama dengan bucket Amazon S3. Nama alias ini dibuat secara otomatis, dan tidak dapat diubah. Nama alias titik akses memenuhi semua persyaratan nama bucket Amazon S3 yang valid dan terdiri dari bagian-bagian berikut:

access point prefix-metadata-s3alias

> **Note**
>
> Sufiks `-s3alias` dicadangkan untuk nama alias titik akses, dan tidak dapat digunakan untuk nama bucket atau titik akses. Untuk informasi selengkapnya tentang penamaan bucket di Amazon S3, lihat Peraturan penamaan bucket.

Kasus penggunaan dan batasan alias titik akses

Saat mengadopsi titik akses, Anda dapat menggunakan alias titik akses tanpa memerlukan perubahan kode yang ekstensif.

Saat Anda membuat titik akses, Amazon S3 secara otomatis menghasilkan nama alias titik akses, seperti yang ditunjukkan dalam contoh berikut. Untuk menjalankan perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control create-access-point --bucket example-s3-bucket1 --name my-access-point --
account-id 111122223333
{
    "AccessPointArn":
    "arn:aws:s3:region:111122223333:accesspoint/my-access-point",
    "Alias": "my-access-point-aqfqprnstn7aefdfbarligizwgyfouse1a-s3alias"
}
```

Anda dapat menggunakan nama alias titik akses ini alih-alih nama bucket Amazon S3 dalam operasi bidang data apa pun. Untuk mengetahui daftar operasi ini, lihat Kompatibilitas titik akses dengan AWS layanan.

AWS CLI Contoh berikut untuk `get-object` perintah menggunakan alias access point bucket untuk mengembalikan informasi tentang objek yang ditentukan. Untuk menjalankan perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3api get-object --bucket my-access-point-aqfqprnstn7aefdfbarligizwgyfouse1a-
s3alias --key dir/my_data.rtf my_data.rtf

{
    "AcceptRanges": "bytes",
    "LastModified": "2020-01-08T22:16:28+00:00",
    "ContentLength": 910,
    "ETag": "\"00751974dc146b76404bb7290f8f51bb\"",
    "VersionId": "null",
    "ContentType": "text/rtf",
    "Metadata": {}
}
```

Batasan

- Alias tidak dapat dikonfigurasi oleh pelanggan.
- Alias tidak dapat dihapus atau dimodifikasi atau dinonaktifkan pada titik akses.
- Anda dapat menggunakan nama alias titik akses ini alih-alih nama bucket Amazon S3 dalam beberapa operasi bidang data. Untuk mengetahui daftar operasi ini, lihat Kompatibilitas titik akses dengan operasi S3.
- Anda tidak dapat menggunakan nama alias titik akses untuk operasi bidang kontrol Amazon S3. Untuk daftar operasi bidang kontrol Amazon S3, lihat Kontrol Amazon S3 di Referensi API Amazon Simple Storage Service.
- Anda tidak dapat menggunakan alias titik akses S3 sebagai sumber atau tujuan untuk operasi Pindah di konsol Amazon S3.
- Alias tidak dapat digunakan dalam kebijakan AWS Identity and Access Management (IAM).
- Alias tidak dapat digunakan sebagai tujuan pencatatan untuk log akses server S3.
- Alias tidak dapat digunakan sebagai tujuan logging untuk AWS CloudTrail log.
- Amazon SageMaker GroundTruth tidak mendukung alias titik akses.

Menggunakan titik akses dengan operasi Amazon S3 yang kompatibel

Contoh berikut ini menunjukkan cara menggunakan titik akses dengan operasi yang kompatibel di Amazon S3.

Topik

- Kompatibilitas titik akses dengan AWS layanan

- Kompatibilitas titik akses dengan operasi S3
- Meminta objek melalui titik akses
- Mengunggah objek melalui alias titik akses
- Menghapus objek melalui titik akses
- Mendaftarkan objek melalui alias titik akses
- Menambahkan rangkaian tag ke objek melalui titik akses
- Memberikan izin akses melalui titik akses menggunakan ACL

Kompatibilitas titik akses dengan AWS layanan

Alias titik akses Amazon S3 memungkinkan aplikasi yang memerlukan nama bucket S3 untuk dengan mudah menggunakan titik akses. Anda dapat menggunakan alias titik akses S3 di mana Anda menggunakan nama bucket S3 untuk mengakses data di S3. Untuk informasi selengkapnya, lihat Kasus penggunaan dan batasan alias titik akses.

Kompatibilitas titik akses dengan operasi S3

Anda dapat menggunakan titik akses untuk mengakses bucket menggunakan subset API Amazon S3 berikut ini. Semua operasi yang tercantum di bawah ini dapat menerima ARN titik akses atau alias titik akses:

Operasi S3

- `AbortMultipartUpload`
- `CompleteMultipartUpload`
- `CopyObject` (hanya salinan wilayah yang sama)
- `CreateMultipartUpload`
- `DeleteObject`
- `DeleteObjectTagging`
- `GetBucketAcl`
- `GetBucketCors`
- `GetBucketLocation`
- `GetBucketNotificationConfiguration`
- `GetBucketPolicy`
- `GetObject`

- GetObjectAcl
- GetObjectAttributes
- GetObjectLegalHold
- GetObjectRetention
- GetObjectTagging
- HeadBucket
- HeadObject
- ListMultipartUploads
- ListObjects
- ListObjectsV2
- ListObjectVersions
- ListParts
- Presign
- PutObject
- PutObjectLegalHold
- PutObjectRetention
- PutObjectAcl
- PutObjectTagging
- RestoreObject
- UploadPart
- UploadPartCopy (hanya salinan wilayah yang sama)

Meminta objek melalui titik akses

Contoh berikut menunjukkan cara meminta objek `my-image.jpg` melalui titik akses `prod` yang dimiliki oleh ID akun `123456789012` di Wilayah `us-west-2`, dan menyimpan file yang telah diunduh sebagai `download.jpg`.

AWS CLI

```
aws s3api get-object --key my-image.jpg --bucket arn:aws:s3:us-
west-2:123456789012:accesspoint/prod download.jpg
```

Mengunggah objek melalui alias titik akses

Contoh berikut ini mengunggah objek `my-image.jpg` melalui alias titik akses `my-access-point-hrzrlukc5m36ft7okagglf3gmwluquse1b-s3alias` yang dimiliki oleh ID akun `123456789012` di Wilayah `us-west-2`.

AWS CLI

```
aws s3api put-object --bucket my-access-point-hrzrlukc5m36ft7okagglf3gmwluquse1b-
s3alias --key my-image.jpg --body my-image.jpg
```

Menghapus objek melalui titik akses

Contoh berikut ini mengunggah objek `my-image.jpg` melalui alias titik akses `prod` yang dimiliki oleh ID akun `123456789012` di Wilayah `us-west-2`.

AWS CLI

```
aws s3api delete-object --bucket arn:aws:s3:us-west-2:123456789012:accesspoint/prod
 --key my-image.jpg
```

Mendaftarkan objek melalui alias titik akses

Contoh berikut mencantumkan objek melalui titik akses alias `my-access-point-hrzrlukc5m36ft7okagglf3gmwluquse1b-s3alias` yang dimiliki oleh ID akun `123456789012` di Wilayah `us-west-2`.

AWS CLI

```
aws s3api list-objects-v2 --bucket my-access-point-
hrzrlukc5m36ft7okagglf3gmwluquse1b-s3alias
```

Menambahkan rangkaian tag ke objek melalui titik akses

Contoh berikut ini menambahkan rangkaian tag ke `my-image.jpg` objek yang sudah ada melalui `prod` titik akses yang dimiliki oleh ID akun `123456789012` di Wilayah `us-west-2`.

AWS CLI

```
aws s3api put-object-tagging --bucket arn:aws:s3:us-west-2:123456789012:accesspoint/
prod --key my-image.jpg --tagging TagSet=[{Key="finance",Value="true"}]
```

Memberikan izin akses melalui titik akses menggunakan ACL

Contoh berikut menerapkan ACL ke `my-image.jpg` objek yang sudah ada melalui `prod` titik akses yang dimiliki oleh ID akun `123456789012` di Wilayah `us-west-2`.

AWS CLI

```
aws s3api put-object-acl --bucket arn:aws:s3:us-west-2:123456789012:accesspoint/prod
 --key my-image.jpg --acl private
```

Pembatasan dan batasan titik akses

Titik akses Amazon S3 memiliki pembatasan dan batasan berikut:

- Setiap titik akses terhubung dengan hanya satu bucket, yang harus Anda tentukan saat membuat titik akses. Setelah Anda membuat titik akses, Anda tidak dapat menghubungkannya dengan bucket yang berbeda. Namun, Anda dapat menghapus titik akses, lalu membuat titik lain dengan nama yang sama dan mengaitkan titik akses baru dengan bucket yang berbeda.
- Nama titik akses harus memenuhi syarat tertentu. Untuk informasi lebih lanjut tentang penamaan titik akses, lihat Aturan penamaan titik akses Amazon S3.
- Setelah Anda membuat titik akses, Anda tidak dapat mengubah konfigurasi cloud privat virtual (VPC).
- Ukuran kebijakan titik akses dibatasi hingga 20 KB.
- Anda dapat membuat maksimum 10.000 titik akses Akun AWS per Wilayah. Jika Anda memerlukan lebih dari 10.000 titik akses untuk satu akun di satu Wilayah, Anda dapat meminta peningkatan kuota layanan. Untuk informasi lebih lanjut tentang kuota layanan dan permintaan peningkatan, lihat AWS Service Quotas dalam Referensi Umum AWS.
- Di Wilayah AWS mana Anda memiliki lebih dari 1.000 titik akses, Anda tidak dapat mencari titik akses berdasarkan nama di konsol Amazon S3.

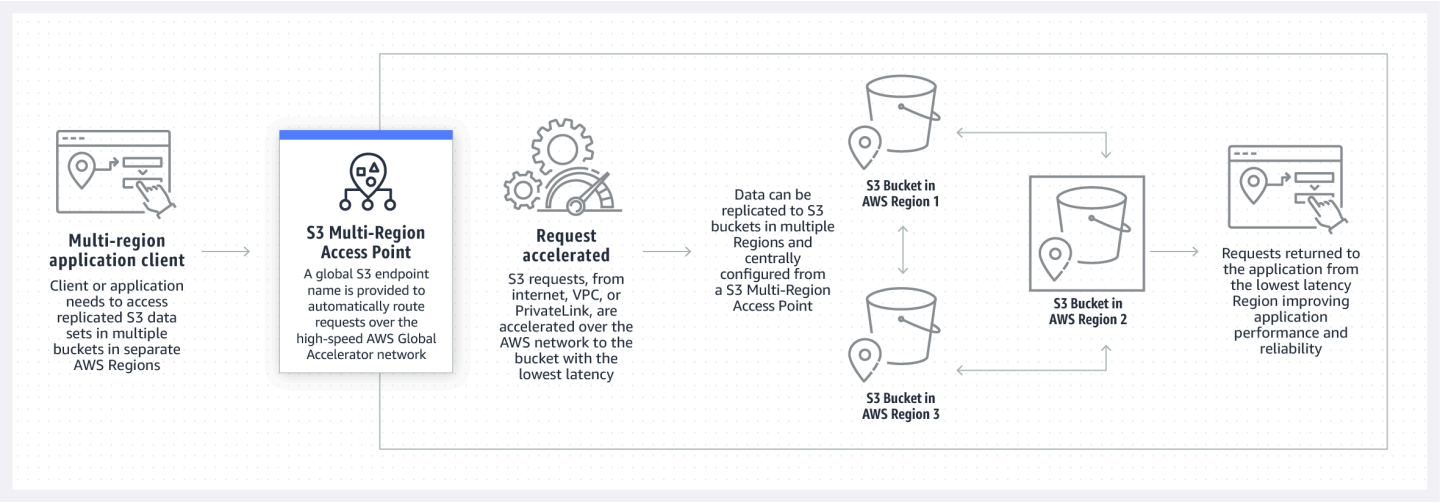
- Anda tidak dapat menggunakan titik akses sebagai tujuan untuk Replikasi S3. Untuk informasi selengkapnya tentang replikasi, lihat Mereplikasi ikhtisar objek.
- Anda tidak dapat menggunakan alias titik akses S3 sebagai sumber atau tujuan untuk operasi Pindah di konsol Amazon S3.
- Anda dapat mengatasi titik akses hanya dengan menggunakan virtual-host-style URL. Untuk informasi lebih lanjut tentang virtual-host-style pengalamatan, lihatMengakses dan mendaftarkan bucket Amazon S3.
- Operasi API yang mengontrol fungsionalitas titik akses (misalnya, `PutAccessPoint` dan `GetAccessPointPolicy`) tidak mendukung panggilan akun silang.
- Anda harus menggunakan AWS Signature Version 4 saat membuat permintaan ke titik akses dengan menggunakan REST API. Untuk informasi selengkapnya tentang mengautentikasi permintaan, lihat Mengautentikasi Permintaan (Versi AWS Tanda Tangan 4) di Referensi API Amazon Simple Storage Service.
- Titik akses hanya mendukung permintaan melalui HTTPS. Amazon S3 akan secara otomatis merespons dengan pengalihan HTTP untuk setiap permintaan yang dibuat melalui HTTP, untuk meningkatkan permintaan ke HTTPS.
- Titik akses tidak mendukung akses anonim.
- Titik akses lintas akun tidak memberikan Anda akses ke data sampai Anda diberikan izin dari pemilik bucket. Pemilik bucket selalu memegang kendali penuh atas akses ke data, dan harus memperbarui kebijakan bucket untuk mengotorisasi permintaan dari titik akses lintas akun. Untuk melihat contoh kebijakan bucket, lihat Mengonfigurasi kebijakan IAM untuk menggunakan titik akses.
- Saat Anda melihat titik akses lintas akun di konsol Amazon S3, kolom Akses menampilkan Tidak Diketahui. Konsol Amazon S3 tidak dapat menentukan apakah akses publik diberikan untuk bucket dan objek terkait. Kecuali Anda memerlukan konfigurasi publik untuk kasus penggunaan tertentu, kami menyarankan agar Anda dan pemilik bucket memblokir semua akses publik ke titik akses dan bucket. Untuk informasi selengkapnya, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

Titik Akses Multi-Wilayah di Amazon S3

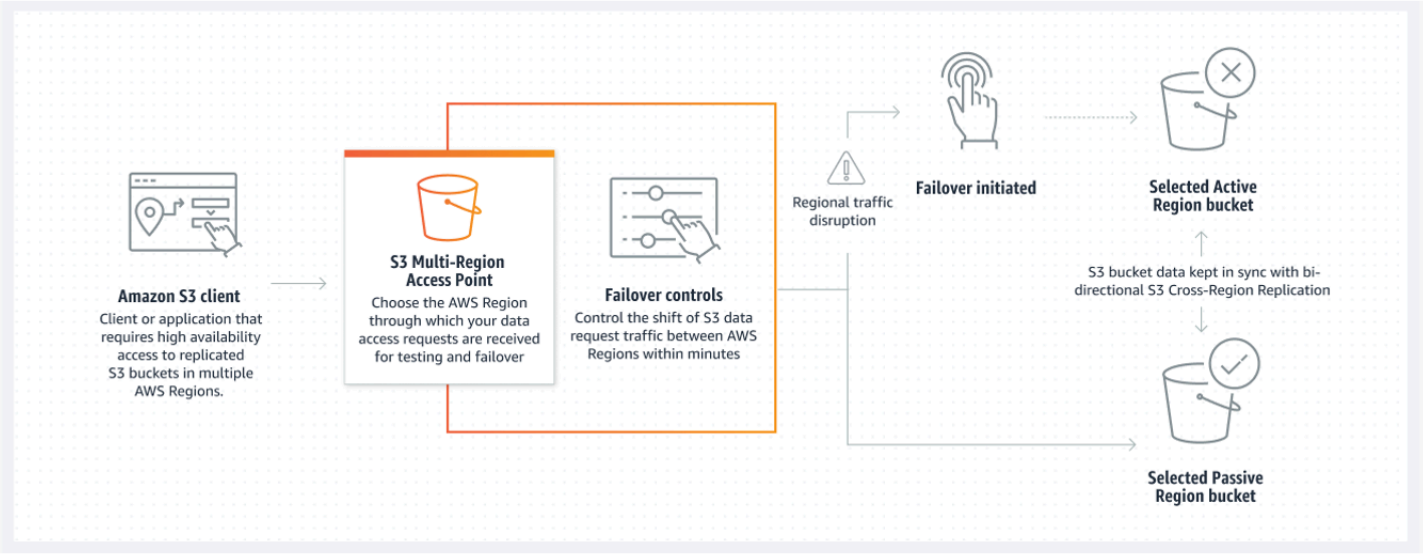
Titik Akses Multi-Wilayah Amazon S3 menyediakan titik akhir global yang dapat digunakan aplikasi untuk memenuhi permintaan dari bucket S3 yang terletak di beberapa Wilayah AWS. Anda dapat menggunakan Titik Akses Multi-Wilayah untuk membangun aplikasi multi-Wilayah dengan arsitektur yang sama dengan yang digunakan di satu Wilayah, dan kemudian menjalankan aplikasi tersebut di mana saja di seluruh dunia. Alih-alih mengirimkan permintaan melalui internet publik yang padat, Titik Akses Multi-Wilayah menyediakan ketahanan jaringan bawaan dengan percepatan permintaan berbasis internet ke Amazon S3. Permintaan aplikasi yang dibuat ke titik akhir global Multi-Region Access Point digunakan AWS Global Acceleratoruntuk secara otomatis merutekan jaringan AWS global ke bucket S3 terdekat dengan status perutean aktif.

Saat Anda membuat Titik Akses Multi-Wilayah, Anda menentukan satu set Wilayah AWS tempat Anda ingin menyimpan data yang akan dilayani melalui Titik Akses Multi-Wilayah tersebut. Anda dapat menggunakan Replikasi Lintas-Wilayah (CRR) S3 untuk menyinkronkan data di antara bucket di Wilayah tersebut. Setelah itu, Anda dapat meminta atau menulis data melalui titik akhir global Titik Akses Multi-Wilayah. Amazon S3 secara otomatis melayani permintaan ke set data yang direplikasi dari Wilayah terdekat yang tersedia. Titik Akses Multi-Wilayah juga kompatibel dengan aplikasi yang berjalan di cloud privat virtual (VPC) Amazon, termasuk yang menggunakan AWS PrivateLink untuk Amazon S3.

Gambar berikut adalah representasi grafis dari Titik Akses Multi-Wilayah Amazon S3 dalam konfigurasi aktif-aktif. Grafik menunjukkan bagaimana permintaan Amazon S3 secara otomatis dialihkan ke bucket di Wilayah AWS aktif terdekat.



Gambar berikut adalah representasi grafis dari Titik Akses Multi-Wilayah Amazon S3 dalam konfigurasi aktif-pasif. Grafik tersebut menunjukkan bagaimana Anda dapat mengontrol lalu lintas akses data Amazon S3 untuk beralih antara Wilayah AWS aktif dan pasif.



Untuk mempelajari selengkapnya tentang cara menggunakan Titik Akses Multi-Wilayah, lihat Tutorial: Memulai dengan Titik Akses Multi-Wilayah Amazon S3.

Topik

- Membuat Titik Akses Multi-Wilayah
- Mengonfigurasi Titik Akses Multi-Wilayah untuk digunakan denganAWS PrivateLink
- Membuat permintaan melalui Titik Akses Multi-Wilayah

Membuat Titik Akses Multi-Wilayah

Untuk membuat Titik Akses Multi-Wilayah di Amazon S3, lakukan hal berikut:

- Tentukan nama untuk Titik Akses Multi-Wilayah.
- Pilih satu bucket di masing-masing Wilayah AWS yang ingin Anda layani permintaan untuk Titik Akses Multi-Wilayah.
- Konfigurasikan pengaturan Blokir Akses Publik Amazon S3 untuk Titik Akses Multi-Wilayah.

Anda memberikan semua informasi ini dalam permintaan buat, yang diproses Amazon S3 secara asinkron. Amazon S3 menyediakan token yang dapat Anda gunakan untuk memantau status permintaan pembuatan asinkron.

Pastikan Anda menyelesaikan peringatan keamanan, kesalahan, peringatan umum, dan saran dari AWS Identity and Access Management Access Analyzer sebelum Anda menyimpan kebijakan Anda. Penganalisis Akses IAM menjalankan pemeriksaan kebijakan untuk memvalidasi kebijakan Anda terhadap tata bahasa kebijakan IAM dan praktik terbaik. Pemeriksaan ini menghasilkan temuan dan memberikan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang fungsional dan sesuai dengan praktik terbaik keamanan. Untuk mempelajari validasi kebijakan menggunakan IAM Access Analyzer lebih lanjut, lihat Validasi kebijakan IAM Access Analyzer di Panduan Pengguna IAM. Untuk melihat daftar peringatan, kesalahan, dan saran yang ditampilkan oleh Penganalisis Akses IAM, lihat referensi pemeriksaan kebijakan Penganalisis Akses IAM.

Saat Anda menggunakan API, permintaan untuk membuat Titik Akses Multi-Wilayah bersifat asinkron. Saat Anda mengirimkan permintaan untuk membuat Titik Akses Multi-Wilayah, Amazon S3 secara sinkron mengotorisasi permintaan tersebut. Kemudian segera memberikan token yang dapat Anda gunakan untuk melacak kemajuan permintaan pembuatan. Untuk informasi selengkapnya tentang pelacakan permintaan asinkron untuk membuat dan mengelola Titik Akses Multi-Wilayah, lihat Menggunakan Titik Akses Multi-Wilayah dengan operasi API yang didukung.

Setelah membuat Titik Akses Multi-Wilayah, Anda dapat membuat kebijakan kontrol akses untuk itu. Setiap Titik Akses Multi-Wilayah dapat memiliki kebijakan terkait. Kebijakan Titik Akses Multi-Wilayah adalah kebijakan berbasis sumber daya yang dapat Anda gunakan untuk membatasi penggunaan Titik Akses Multi-Wilayah berdasarkan sumber daya, pengguna, atau kondisi lainnya.

> **Note**
>
> Agar aplikasi atau pengguna dapat mengakses objek melalui Titik Akses Multi-Wilayah, kedua kebijakan berikut ini harus mengizinkan permintaan tersebut:
>
> - Kebijakan akses untuk Titik Akses Multi-Wilayah
> - Kebijakan akses untuk bucket dasar yang berisi objek tersebut
>
> Jika kedua kebijakan tersebut berbeda, maka kebijakan yang lebih ketat yang diutamakan.

Untuk menyederhanakan pengelolaan izin untuk Titik Akses Multi-Wilayah, Anda dapat mendelegasikan kontrol akses dari bucket ke Titik Akses Multi-Wilayah. Untuk informasi selengkapnya, lihat the section called “Contoh kebijakan Multi-Region Access Point”.

Penggunaan bucket dengan Titik Akses Multi-Wilayah tidak akan mengubah perilaku bucket tersebut ketika diakses melalui nama bucket yang sudah ada atau melalui Amazon Resource Name (ARN). Semua operasi yang ada terhadap bucket akan terus bekerja seperti sebelumnya. Pembatasan yang Anda sertakan dalam kebijakan Titik Akses Multi-Wilayah hanya berlaku untuk permintaan yang dibuat melalui Titik Akses Multi-Wilayah.

Anda dapat memperbarui kebijakan untuk Titik Akses Multi-Wilayah setelah dibuat, tetapi tidak dapat menghapus kebijakan tersebut. Namun, Anda dapat memperbarui kebijakan Titik Akses Multi-Wilayah untuk menolak semua izin.

Topik

- Aturan penamaan Titik Akses Multi-Wilayah Amazon S3
- Aturan dalam memilih bucket untuk Titik Akses Multi-Wilayah Amazon S3
- Membuat Titik Akses Multi-Wilayah Amazon S3
- Memblokir akses publik dengan Titik Akses Multi-Wilayah Amazon S3
- Melihat detail konfigurasi Titik Akses Multi-Wilayah Amazon S3
- Menghapus Titik Akses Multi-Wilayah

Aturan penamaan Titik Akses Multi-Wilayah Amazon S3

Saat membuat Titik Akses Multi-Wilayah, Anda memberinya nama, yang merupakan string yang Anda pilih. Anda tidak dapat mengubah nama Titik Akses Multi-Region setelah dibuat. Nama harus unik dalam Akun AWS Anda, dan harus sesuai dengan persyaratan penamaan yang tercantum di Pembatasan dan batasan Titik Akses Multi-Wilayah. Untuk membantu Anda mengidentifikasi Titik Akses Multi-Wilayah, gunakan nama yang memiliki arti penting bagi Anda, organisasi, atau yang mencerminkan skenario.

Anda menggunakan nama ini saat menginvokasi operasi manajemen Titik Akses Multi-Wilayah, seperti `GetMultiRegionAccessPoint` dan `PutMultiRegionAccessPointPolicy`. Nama ini tidak digunakan untuk mengirim permintaan ke Titik Akses Multi-Wilayah, dan tidak perlu diekspos ke klien yang membuat permintaan dengan menggunakan Titik Akses Multi-Wilayah.

Saat membuat Titik Akses Multi-Wilayah, Amazon S3 secara otomatis memberikan nama alias untuk titik akses tersebut. Alias ini adalah string alfanumerik unik yang diakhiri dengan `.mrap`. Alias digunakan untuk membangun nama host dan Amazon Resource Name (ARN) untuk Titik Akses Multi-Wilayah. Nama yang sepenuhnya memenuhi syarat juga didasarkan pada alias untuk Titik Akses Multi-Wilayah.

Anda tidak dapat menentukan nama Titik Akses Multi-Wilayah dari nama aliasnya, sehingga Anda dapat mengungkapkan nama alias tanpa risiko mengungkap nama, tujuan, atau pemilik Titik Akses Multi-Wilayah. Amazon S3 memilih alias untuk setiap Titik Akses Multi-Wilayah baru, dan alias tidak dapat diubah. Untuk informasi selengkapnya tentang menangani Titik Akses Multi-Wilayah, lihat Membuat permintaan melalui Titik Akses Multi-Wilayah.

Alias Titik Akses Multi-Wilayah bersifat unik sepanjang waktu dan tidak didasarkan pada nama atau konfigurasi Titik Akses Multi-Wilayah. Jika Anda membuat Titik Akses Multi-Wilayah, lalu menghapusnya dan membuat yang lain dengan nama dan konfigurasi yang sama, Titik Akses Multi-Wilayah kedua akan memiliki alias yang berbeda dari yang pertama. Titik Akses Multi-Wilayah baru tidak akan pernah memiliki alias yang sama dengan Titik Akses Multi-Wilayah sebelumnya.

Aturan dalam memilih bucket untuk Titik Akses Multi-Wilayah Amazon S3

Setiap Titik Akses Multi-Wilayah dikaitkan dengan Wilayah tempat Anda ingin memenuhi permintaan. Titik Akses Multi-Wilayah harus dikaitkan dengan hanya satu bucket di masing-masing Wilayah tersebut. Anda menentukan nama setiap bucket dalam permintaan untuk membuat Titik Akses Multi-Wilayah. Bucket yang mendukung Titik Akses Multi-Region bisa sama dengan Akun AWS yang memiliki Titik Akses Multi-Region, atau bisa juga di tempat lain. Akun AWS

Satu bucket dapat digunakan oleh beberapa Titik Akses Multi-Wilayah.

> **Important**
>
> - Anda dapat menentukan bucket yang terkait dengan Titik Akses Multi-Wilayah hanya pada saat Anda membuatnya. Setelah dibuat, Anda tidak dapat menambahkan, memodifikasi, atau menghapus bucket dari konfigurasi Titik Akses Multi-Wilayah. Untuk mengubah bucket, Anda harus menghapus seluruh Titik Akses Multi-Wilayah dan membuat titik akses baru.
> - Anda tidak dapat menghapus bucket yang merupakan bagian dari Titik Akses Multi-Wilayah. Jika Anda ingin menghapus bucket yang dilampirkan ke Titik Akses Multi-Wilayah, hapus Titik Akses Multi-Wilyah terlebih dahulu.

- Jika Anda menambahkan bucket yang dimiliki oleh akun lain ke Titik Akses Multi-Wilayah, pemilik bucket juga harus memperbarui kebijakan bucket untuk memberikan izin akses ke Titik Akses Multi-Wilayah. Jika tidak, Titik Akses Multi-Wilayah tidak akan dapat mengambil data dari bucket tersebut. Misalnya kebijakan yang menunjukkan cara memberikan akses tersebut, lihat contoh kebijakan Titik Akses Multi-Wilayah.
- Tidak semua Wilayah mendukung Titik Akses Multi-Wilayah. Untuk melihat daftar Wilayah yang didukung, lihat Pembatasan dan batasan Titik Akses Multi-Wilayah.

Anda dapat membuat aturan replikasi untuk menyinkronkan data antar bucket. Dengan aturan ini, Anda dapat menyalin data secara otomatis dari bucket sumber ke bucket tujuan. Menghubungkan bucket ke Titik Akses Multi-Wilayah tidak akan memengaruhi cara kerja replikasi. Untuk mengonfigurasi replikasi dengan Titik Akses Multi-Wilayah akan dijelaskan di bagian selanjutnya.

> **Important**
>
> Ketika Anda membuat permintaan ke Titik Akses Multi-Wilayah, Titik Akses Multi-Wilayah tidak mengetahui isi data bucket di dalamnya. Oleh karena itu, bucket yang mendapatkan permintaan mungkin tidak berisi data yang diminta. Untuk membuat set data yang konsisten di bucket Amazon S3 yang terkait dengan Titik Akses Multi-Wilayah, sebaiknya Anda mengonfigurasi Replikasi Lintas Wilayah (CRR) S3. Untuk informasi selengkapnya, lihat Mengkonfigurasi replikasi untuk digunakan dengan Titik Akses Multi-Wilayah.

Membuat Titik Akses Multi-Wilayah Amazon S3

Contoh berikut menunjukkan cara membuat Titik Akses Multi-Wilayah dengan menggunakan konsol Amazon S3.

Menggunakan konsol S3

Untuk membuat Titik Akses Multi-Wilayah

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Titik Akses Multi-Wilayah.
3. Pilih Buat Titik Akses Multi-Wilayah untuk mulai membuat Titik Akses Multi-Wilayah Anda.

4. Pada halaman Titik Akses Multi-Wilayah, berikan nama untuk Titik Akses Multi-Wilayah di bidang nama Titik Akses Multi-Wilayah.
5. Pilih bucket yang akan dikaitkan dengan Titik Akses Multi-Wilayah ini. Anda dapat memilih bucket yang ada di akun Anda, atau pilih bucket dari akun lain.

> **Note**
>
> Anda harus menambahkan setidaknya satu bucket dari akun Anda atau akun lain. Selain itu, ketahuilah bahwa Titik Akses Multi-Wilayah hanya mendukung satu bucket per Wilayah AWS. Oleh karena itu, Anda tidak dapat menambahkan dua bucket dari Wilayah yang sama. Wilayah AWS yang dinonaktifkan secara default tidak didukung.

- Untuk menambahkan bucket di akun Anda, pilih Tambahkan bucket. Daftar semua bucket di akun Anda akan ditampilkan. Anda dapat mencari bucket berdasarkan nama, atau mengurutkan nama bucket berdasarkan urutan abjad.
- Untuk menambahkan bucket dari akun lain, pilih Tambahkan bucket dari akun lain. Pastikan Anda mengetahui nama bucket dan Akun AWS ID yang tepat karena Anda tidak dapat mencari atau menelusuri bucket di akun lain.

> **Note**
>
> Anda harus memasukkan Akun AWS ID dan nama bucket yang valid. Bucket juga harus berada di Wilayah yang didukung, jika tidak Anda akan mengalami kesalahan saat mencoba membuat Titik Akses Multi-Wilayah. Untuk daftar Wilayah yang mendukung Titik Akses Multi-Wilayah, lihat Pembatasan dan batasan Titik Akses Multi-Wilayah.

6. (Opsional) Jika Anda ingin menghapus bucket yang telah Anda tambahkan, pilih Hapus.

> **Note**
>
> Anda tidak dapat menambahkan atau menghapus bucket ke Titik Akses Multi-Wilayah ini setelah Anda selesai membuatnya.

7. Di Bawah Pengaturan Blokir Akses Publik untuk Titik Akses Multi-Wilayah, pilih pengaturan Blokir Akses Publik yang ingin Anda terapkan ke Titik Akses Multi-Wilayah. Secara default,

semua pengaturan Blokir Akses Publik diaktifkan untuk Titik Akses Multi-Wilayah baru. Sebaiknya aktifkan semua pengaturan kecuali jika Anda tahu bahwa ada tujuan khusus untuk menonaktifkannya.

> **Note**
>
> Anda tidak dapat mengubah pengaturan Blokir Akses Publik untuk Titik Akses Multi-Wilayah setelah Titik Akses Multi-Wilayah dibuat. Oleh karena itu, jika Anda ingin memblokir akses publik, pastikan aplikasi Anda berfungsi dengan benar tanpa akses publik sebelum Anda membuat Titik Akses Multi-Wilayah.

8. Pilih Buat Titik Akses Multi-Wilayah.

> **Important**
>
> Saat Anda menambahkan bucket milik akun lain ke Titik Akses Multi-Wilayah, pemilik bucket juga harus memperbarui kebijakan bucket untuk memberikan izin akses ke Titik Akses Multi-Wilayah. Jika tidak, Titik Akses Multi-Wilayah tidak akan dapat mengambil data dari bucket tersebut. Misalnya kebijakan yang menunjukkan cara memberikan akses tersebut, lihat contoh kebijakan Titik Akses Multi-Wilayah.

Menggunakan AWS CLI

Anda dapat menggunakan AWS CLI untuk membuat Titik Akses Multi-Region. Saat Anda membuat Titik Akses Multi-Wilayah, Anda harus menyediakan semua bucket yang akan didukungnya. Anda tidak dapat menambahkan bucket ke Titik Akses Multi-Wilayah setelah dibuat.

Berikut ini contoh membuat Titik Akses Multi-Wilayah dengan dua bucket menggunakan perintah AWS CLI. Untuk menggunakan perintah contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control create-multi-region-access-point --account-id 111122223333 --details '{
        "Name": "simple-multiregionaccesspoint-with-two-regions",
        "PublicAccessBlock": {
            "BlockPublicAcls": true,
            "IgnorePublicAcls": true,
            "BlockPublicPolicy": true,
            "RestrictPublicBuckets": true
```

```
        },
        "Regions": [
            { "Bucket": "example-s3-bucket1" },
            { "Bucket": "example-s3-bucket2" }
        ]
    }' --region us-west-2
```

Memblokir akses publik dengan Titik Akses Multi-Wilayah Amazon S3

Setiap Titik Akses Multi-Wilayah memiliki pengaturan berbeda untuk Blokir Akses Publik Amazon S3. Pengaturan ini beroperasi bersama dengan pengaturan Blokir Akses Publik untuk Akun AWS yang memiliki Titik Akses Multi-Wilayah dan bucket yang mendasarinya.

Ketika mengotorisasi permintaan, Amazon S3 akan menerapkan kombinasi yang paling ketat dari pengaturan ini. Jika pengaturan Blokir Akses Publik untuk salah satu sumber daya ini (akun pemilik Titik Akses Multi-Wilayah, bucket yang mendasari, atau akun pemilik bucket) memblokir akses untuk tindakan atau sumber daya yang diminta, maka Amazon S3 akan menolak permintaan tersebut.

Sebaiknya aktifkan semua pengaturan Blokir Akses Publik kecuali jika ada kebutuhan khusus untuk menonaktifkan salah satu dari pengaturan tersebut. Secara default, semua pengaturan Blokir Akses Publik diaktifkan untuk Titik Akses Multi-Wilayah. Jika Blokir Akses Publik diaktifkan, Titik Akses Multi-Wilayah tidak dapat menerima permintaan berbasis internet.

> **Important**
>
> Anda tidak dapat mengubah pengaturan Blokir Akses Publik untuk Titik Akses Multi-Wilayah setelah dibuat.

Untuk informasi selengkapnya tentang Blokir Akses Publik Amazon S3, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

Melihat detail konfigurasi Titik Akses Multi-Wilayah Amazon S3

Berikut ini adalah contoh yang menunjukkan cara melihat detail konfigurasi Titik Akses Multi-Wilayah dengan menggunakan konsol Amazon S3.

Menggunakan konsol S3

Untuk membuat Titik Akses Multi-Wilayah

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Titik Akses Multi-Wilayah.
3. Pilih nama Titik Akses Multi-Wilayah yang ingin Anda lihat detail konfigurasinya.

   - Tab Properti mencantumkan semua bucket yang terkait dengan Titik Akses Multi-Wilayah, tanggal pembuatan, Amazon Resource Name (ARN), dan alias. Kolom ID Akun AWS juga mencantumkan setiap bucket yang dimiliki oleh akun eksternal yang terkait dengan Titik Akses Multi-Wilayah Anda.
   - Tab Izin mencantumkan pengaturan Blokir Akses Publik yang diterapkan ke bucket yang terkait dengan Titik Akses Multi-Wilayah ini. Anda juga dapat melihat kebijakan Titik Akses Multi-Wilayah untuk Titik Akses Multi-Wilayah yang telah anda buat sebelumnya. Notifikasi Info di halaman Izin juga mencantumkan semua bucket (di akun Anda dan akun lain) untuk Titik Akses Multi-Wilayah ini yang mengaktifkan pengaturan Akses Publik diblokir.
   - Tab Replikasi dan failover menyediakan tampilan peta dari bucket yang terkait dengan Titik Akses Multi-Wilayah Anda dan Wilayah tempat bucket berada. Jika ada bucket dari akun lain yang tidak memiliki izin untuk menarik data, Wilayah akan ditandai dengan warna merah di peta Ringkasan replikasi, yang menunjukkan bahwa itu adalah Wilayah AWS dengan kesalahan mendapatkan status replikasi.

     > **Note**
     >
     > Untuk mengambil informasi status replikasi dari bucket di akun eksternal, pemilik bucket harus memberi Anda izin `s3:GetBucketReplication` dalam kebijakan bucket mereka.

     Tab ini juga menyediakan metrik replikasi, aturan replikasi, dan status failover untuk Wilayah yang digunakan dengan Titik Akses Multi-Wilayah Anda.

Menggunakan AWS CLI

Anda dapat menggunakan AWS CLI untuk melihat detail konfigurasi untuk Titik Akses Multi-Wilayah.

AWS CLI Contoh berikut mendapatkan konfigurasi Titik Akses Multi-Region Anda saat ini. Untuk menggunakan contoh perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control get-multi-region-access-point --account-id 111122223333 --name example-
s3-bucket1
```

Menghapus Titik Akses Multi-Wilayah

Prosedur berikut menjelaskan cara menghapus Titik Akses Multi-Wilayah dengan menggunakan konsol Amazon S3.

Menghapus Titik Akses Multi-Wilayah tidak akan menghapus bucket yang terkait dengan Titik Akses Multi-Wilayah, tetapi hanya Titik Akses Multi-Wilayah itu sendiri.

Menggunakan konsol S3

Untuk menghapus Titik Akses Multi-Wilayah

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Titik Akses Multi-Wilayah.
3. Pilih tombol opsi di samping nama Titik Akses Multi-Wilayah Anda.
4. Pilih Hapus.
5. Di kotak dialog Hapus Titik Akses Multi-Wilayah, masukkan nama AWS bucket yang ingin Anda hapus.

   > **Note**
   >
   > Pastikan untuk memasukkan nama bucket yang valid. Jika tidak, tombol Hapus akan dinonaktifkan.

6. Pilih Hapus untuk mengonfirmasi penghapusan Titik Akses Multi-Wilayah Anda.

Menggunakan AWS CLI

Anda dapat menggunakan AWS CLI untuk menghapus Titik Akses Multi-Region. Tindakan ini tidak menghapus bucket yang terkait dengan Titik Akses Multi-Wilayah, tetapi hanya Titik Akses Multi-

Wilayah itu sendiri. Untuk menggunakan contoh perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control delete-multi-region-access-point --account-id 123456789012 --details
 Name=example-multi-region-access-point-name
```

Mengonfigurasi Titik Akses Multi-Wilayah untuk digunakan denganAWS PrivateLink

Anda dapat menggunakan Titik Akses Multi-Wilayah untuk merutekan lalu lintas permintaan Amazon S3 di antaranyaWilayah AWS. Setiap titik akhir global Titik Akses Multi-Wilayah merutekan lalu lintas permintaan data Amazon S3 dari berbagai sumber tanpa Anda harus membangun konfigurasi jaringan yang kompleks dengan titik akhir yang terpisah. Sumber lalu lintas permintaan data ini meliputi:

- Lalu lintas yang berasal dari virtual private cloud (VPC)
- Lalu lintas dari pusat data lokal yang bepergianAWS PrivateLink
- Lalu lintas dari internet publik

Jika Anda membuatAWS PrivateLink sambungan ke Titik Akses Multi-Wilayah S3, Anda dapat merutekan permintaan S3 keAWS, atau di beberapaWilayah AWS, melalui koneksi pribadi dengan menggunakan arsitektur dan konfigurasi jaringan sederhana. Saat Anda mengonfigurasiAWS PrivateLink, Anda tidak perlu mengonfigurasi

Topik

- Mengonfigurasi Titik Akses Multi-Wilayah untuk digunakan denganAWS PrivateLink
- Menghapus akses ke Titik Akses Multi-Wilayah dari titik akhir VPC

Mengonfigurasi Titik Akses Multi-Wilayah untuk digunakan denganAWS PrivateLink

AWS PrivateLinkmemberi Anda konektivitas pribadi ke Amazon S3 menggunakan alamat IP pribadi di cloud pribadi virtual (VPC) Anda. Anda dapat menyediakan satu atau beberapa titik akhir antarmuka di dalam VPC Anda untuk terhubung ke Titik Akses Multi-Wilayah Amazon S3.

Anda dapat membuat titik akhir com.amazonaws.s3-global.accesspoint untuk Titik Akses Multi-Wilayah melaluiAWS Management Console,AWS CLI, atauAWS SDK. Untuk mempelajari selengkapnya tentang cara mengkonfigurasi titik akhir antarmuka untuk Titik Akses Multi-Wilayah, lihat Endpoint Antarmuka VPC di Panduan Pengguna VPC.

Untuk membuat permintaan ke Titik Akses Multi-Wilayah melalui titik akhir antarmuka, ikuti langkah-langkah ini untuk mengonfigurasi VPC dan Titik Akses Multi-Wilayah.

Untuk mengonfigurasi Titik Akses Multi-Wilayah yang akan digunakanAWS PrivateLink

1. Buat atau miliki endpoint VPC yang sesuai yang dapat terhubung ke Titik Akses Multi-Wilayah. Untuk informasi selengkapnya tentang membuat titik akhir VPC, lihat Endpoint Antarmuka VPC di Panduan Pengguna VPC.

   > **Important**
   >
   > Pastikan untuk membuat endpoint com.amazonaws.s3-global.accesspoint. Tipe endpoint lainnya tidak dapat mengakses Titik Akses Multi-Wilayah.

   Setelah titik akhir VPC ini dibuat, semua permintaan Titik Akses Multi-Wilayah di rute VPC melalui titik akhir ini jika Anda mengaktifkan DNS pribadi untuk titik akhir. Ini tidak diaktifkan secara default.
2. Jika kebijakan Multi-Region Access Point tidak mendukung koneksi dari titik akhir VPC, Anda harus memperbaruinya.
3. Verifikasi bahwa kebijakan bucket individual akan memungkinkan akses ke pengguna Titik Akses Multi-Wilayah.

Ingatlah bahwa Titik Akses Multi-Wilayah berfungsi dengan merutekan permintaan ke bucket, bukan dengan memenuhi permintaan itu sendiri. Hal ini penting untuk diingat karena pencetus permintaan harus memiliki izin ke Multi-Region Access Point dan diizinkan untuk mengakses bucket individu di Multi-Region Access Point. Jika tidak, permintaan mungkin dialihkan ke bucket di mana pencetus tidak memiliki izin untuk memenuhi permintaan. Titik Akses Multi-Wilayah dan bucket yang terkait dapat dimiliki olehAWS akun yang sama atau lainnya. Namun, VPC dari akun yang berbeda dapat menggunakan Titik Akses Multi-Wilayah jika izin dikonfigurasi dengan benar.

Karena itu, kebijakan endpoint VPC harus mengizinkan akses ke Titik Akses Multi-Wilayah dan ke setiap bucket dasar yang Anda inginkan untuk dapat memenuhi permintaan. Sebagai contoh,

anggaplah Anda memiliki Multi-Region Access Point dengan aliasmfzwi23gnjvgw.mrap. Ini didukung oleh bucketDOC-EXAMPLE-BUCKET1 danDOC-EXAMPLE-BUCKET2, semuanya dimiliki oleh akun AWS123456789012. Dalam hal ini, kebijakan endpoint VPC berikut akan memungkinkanGetObject permintaan dari VPC yang dibuatmfzwi23gnjvgw.mrap untuk dipenuhi oleh backing bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Sid": "Read-buckets-and-MRAP-VPCE-policy",
        "Principal": "*",
        "Action": [
            "s3:GetObject"
        ],
        "Effect": "Allow",
        "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*",
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET2/*",
            "arn:aws:s3::123456789012:accesspoint/mfzwi23gnjvgw.mrap/object/*"
        ]
    }]
}
```

Seperti disebutkan sebelumnya, Anda juga harus memastikan bahwa kebijakan Multi-Region Access Point dikonfigurasi untuk mendukung akses melalui titik akhir VPC. Anda tidak perlu menentukan titik akhir VPC yang meminta akses. Kebijakan contoh berikut akan memberikan akses ke pemohon yang mencoba menggunakan Titik Akses Multi-Wilayah untukGetObject permintaan tersebut.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Sid": "Open-read-MRAP-policy",
        "Effect": "Allow",
        "Principal": "*",
        "Action": [
            "s3:GetObject"
          ],
        "Resource": "arn:aws:s3::123456789012:accesspoint/mfzwi23gnjvgw.mrap/object/*"
    }]
```

```
}
```

Dan tentu saja, masing-masing bucket akan membutuhkan kebijakan untuk mendukung akses dari permintaan yang dikirimkan melalui endpoint VPC. Kebijakan contoh berikut memberikan akses baca ke setiap pengguna anonim, yang akan mencakup permintaan yang dibuat melalui titik akhir VPC.

```
{
    "Version":"2012-10-17",
   "Statement": [
   {
       "Sid": "Public-read",
       "Effect":"Allow",
       "Principal": "*",
       "Action": "s3:GetObject",
       "Resource": [
           "arn:aws:s3:::DOC-EXAMPLE-BUCKET1",
           "arn:aws:s3:::DOC-EXAMPLE-BUCKET2/*"]
    }]
}
```

Untuk informasi selengkapnya tentang mengedit kebijakan endpoint VPC, lihat Mengontrol akses ke layanan dengan titik akhir VPC di Panduan Pengguna VPC.

Menghapus akses ke Titik Akses Multi-Wilayah dari titik akhir VPC

Jika Anda memiliki Titik Akses Multi-Wilayah dan ingin menghapus akses ke titik akhir antarmuka, Anda harus menyediakan kebijakan akses baru untuk Titik Akses Multi-Wilayah yang mencegah akses untuk permintaan yang datang melalui titik akhir VPC. Namun, jika bucket di Multi-Region Access Point Anda mendukung permintaan melalui titik akhir VPC, mereka akan terus mendukung permintaan ini. Jika Anda ingin mencegah dukungan itu, Anda juga harus memperbarui kebijakan untuk bucket. Memasok kebijakan akses baru ke Multi-Region Access Point mencegah akses hanya ke Multi-Region Access Point, bukan ke bucket yang mendasarinya.

> **Note**
>
> Anda tidak dapat menghapus kebijakan akses untuk Titik Akses Multi-Wilayah. Untuk menghapus akses ke Titik Akses Multi-Wilayah, Anda harus memberikan kebijakan akses baru dengan akses yang dimodifikasi yang Anda inginkan.

Alih-alih memperbarui kebijakan akses untuk Titik Akses Multi-Wilayah, Anda dapat memperbarui kebijakan bucket untuk mencegah permintaan melalui titik akhir VPC. Dalam hal ini, pengguna masih dapat mengakses Titik Akses Multi-Wilayah melalui titik akhir VPC. Namun, jika permintaan Titik Akses Multi-Wilayah dialihkan ke bucket tempat kebijakan bucket mencegah akses, permintaan akan menghasilkan pesan kesalahan.

Membuat permintaan melalui Titik Akses Multi-Wilayah

Seperti sumber daya lain, Amazon S3 Titik akses Multi-Region memiliki Amazon Resource Names (ARN). Anda dapat menggunakan ARN ini untuk mengarahkan permintaan ke Titik Akses Multi-Wilayah dengan menggunakanAWS Command Line Interface (AWS CLI),AWS SDK, atau API Amazon S3. Anda juga dapat menggunakan ARN ini untuk mengidentifikasi Titik Akses Multi-Wilayah dalam kebijakan kontrol akses. Sebuah Multi-Region Access Point ARN tidak menyertakan atau mengungkapkan nama Multi-Region Access Point. Untuk informasi lebih lanjut tentang ARN, lihat Amazon Resource Names (ARN) di ARN Referensi Umum AWS.

> **Note**
>
> Alias Multi-Region Access Point dan ARN tidak dapat digunakan secara bergantian.

ARN Titik akses Multi-wilayah menggunakan format berikut:

`arn:aws:s3::`*`account-id`*`:accesspoint/`*`MultiRegionAccessPoint_alias`*

Berikut ini adalah beberapa contoh dari Multi-Region Access Point ARN:

- `arn:aws:s3::123456789012:accesspoint/mfzwi23gnjvgw.mrap`mewakili Multi-Region Access Point dengan alias`mfzwi23gnjvgw.mrap`, yang dimiliki oleh`Akun AWS123456789012`.
- `arn:aws:s3::123456789012:accesspoint/*`mewakili semua Titik Akses Multi-Wilayah di bawah akun`123456789012`. ARN ini cocok dengan semua Titik Akses Multi-Wilayah untuk akun`123456789012`, tetapi tidak cocok dengan Titik Akses Amazon S3 Regional apa pun karena ARN tidak menyertakan fileWilayah AWS. Sebaliknya, ARN`arn:aws:s3:us-west-2:123456789012:accesspoint/*` cocok dengan semua Titik Akses Amazon S3 Regional di Wilayah`us-west-2` untuk akun`123456789012`, tetapi tidak cocok dengan Titik Akses Multi-Wilayah mana pun.

ARN untuk objek yang diakses melalui Multi-Region Access Point menggunakan format berikut:

`arn:aws:s3::account_id:accesspoint/MultiRegionAccessPoint_alias//key`

Seperti dengan Multi-Region Access Point ARN, ARN untuk objek yang diakses melalui Multi-Region Access Point tidak menyertakanWilayah AWS. Berikut ini adalah beberapa contoh.

- `arn:aws:s3::123456789012:accesspoint/mfzwi23gnjvgw.mrap//-01`mewakili`-01`, yang diakses melalui Multi-Region Access Point dengan alias`mfzwi23gnjvgw.mrap`, yang dimiliki oleh akun`123456789012`.
- `arn:aws:s3::123456789012:accesspoint/mfzwi23gnjvgw.mrap//*`mewakili semua objek yang dapat diakses melalui Multi-Region Access Point dengan alias`mfzwi23gnjvgw.mrap`, di akun`123456789012`.
- `arn:aws:s3::123456789012:accesspoint/mfzwi23gnjvgw.mrap//-01/finance/*`mewakili semua objek yang dapat diakses di bawah`-01/finance/` untuk Multi-Region Access Point dengan alias`mfzwi23gnjvgw.mrap`, di akun`123456789012`.

Nama host Titik Akses Multi-Wilayah

Anda dapat mengakses data di Amazon S3 melalui Titik Akses Multi-Wilayah dengan menggunakan nama host dari Titik Akses Multi-Wilayah. Permintaan dapat diarahkan ke hostname ini dari internet publik. Jika Anda telah mengonfigurasi satu atau lebih gateway internet untuk Titik Akses Multi-Wilayah, permintaan juga dapat diarahkan ke nama host ini dari cloud pribadi virtual (VPC). Untuk informasi selengkapnya tentang membuat titik akhir antarmuka VPC untuk digunakan dengan Titik Akses Multi-Wilayah, lihat[Mengonfigurasi Titik Akses Multi-Wilayah untuk digunakan denganAWS PrivateLink](#).

Untuk membuat permintaan melalui Multi-Region Access Point dari VPC dengan menggunakan endpoint VPC, Anda dapat menggunakannyaAWS PrivateLink. Saat Anda membuat permintaan ke Titik Akses Multi-Wilayah dengan menggunakanAWS PrivateLink, Anda tidak dapat langsung menggunakan nama Regional DOMAIN NAME SYSTEM (DNS) khusus titik akhir yang diakhiri dengan`region.vpce.amazonaws.com`. Nama host ini tidak akan memiliki sertifikat yang terkait dengannya, sehingga tidak dapat digunakan secara langsung. Anda masih dapat menggunakan nama DOMAIN NAME SYSTEM (DNS) publik dari titik akhir VPC sebagai`CNAME` atau`ALIAS` target. Atau, Anda dapat mengaktifkan sistem nama domain pribadi (DNS) pada titik akhir dan menggunakan nama sistem nama`MultiRegionAccessPoint_alias.accesspoint.s3-global.amazonaws.com` domain (DNS) titik akses multi-wilayah standar, seperti yang dijelaskan di bagian ini.

Saat Anda mengajukan permintaan ke API untuk operasi data Amazon S3 (misalnya,`GetObject`) melalui Titik Akses Multi-Wilayah, nama host untuk permintaan tersebut adalah sebagai berikut:

`MultiRegionAccessPoint_alias``.accesspoint.s3-global.amazonaws.com`

Misalnya, untuk membuat`GetObject` permintaan melalui Multi-Region Access Point dengan alias`mfzwi23gnjvgw.mrap`, buat permintaan ke hostname`mfzwi23gnjvgw.mrap.accesspoint.s3-global.amazonaws.com`. `s3-global`Bagian dari hostname menunjukkan bahwa hostname ini bukan untuk Region tertentu.

Membuat permintaan melalui Multi-Region Access Point mirip dengan membuat permintaan melalui jalur akses Single-region. Namun, penting untuk mengetahui perbedaan berikut ini:

- Multi-Region Access Point ARN tidak menyertakan fileWilayah AWS. Mereka mengikuti formatnya`arn:aws:s3::`*`account-id`*`:accesspoint/`*`MultiRegionAccessPoint_alias`*.
- Untuk permintaan yang dibuat melalui operasi API (permintaan ini tidak memerlukan penggunaan ARN), Titik Akses Multi-Wilayah menggunakan skema titik akhir yang berbeda. Skema ini*`MultiRegionAccessPoint_alias`*`.accesspoint.s3-global.amazonaws.com` — misalnya,`mfzwi23gnjvgw.mrap.accesspoint.s3-global.amazonaws.com`. Perhatikan perbedaan dibandingkan dengan titik akses Single-region:
  - Nama host Titik Akses Multi-Wilayah menggunakan alias mereka, bukan nama Titik Akses Multi-Wilayah.
  - Nama host Titik Akses Multi-Wilayah tidak menyertakanAkun AWS ID pemilik.
  - Nama host Titik Akses Multi-Wilayah tidak menyertakan nama hostWilayah AWS.
  - Nama host Multi-Region Access Point termasuk`s3-global.amazonaws.com` sebagai gantinya`s3.amazonaws.com`.
- Permintaan Titik Akses Multi-Wilayah harus ditandatangani dengan menggunakan Signature Version 4A (SigV4a). Saat Anda menggunakanAWS SDK, SDK secara otomatis mengonversi Sigv4 ke Sigv4a. Oleh karena itu, pastikan AWSSDK Anda mendukung SigV4a sebagai implementasi penandatanganan yang digunakan untuk menandatanganiWilayah AWS permintaan global. Untuk informasi selengkapnya tentang Sigv4a, lihat Menandatangani permintaanAWS API di Referensi Umum AWS.

Titik akses Multi-wilayah dan Amazon S3 Transfer Acceleration

Amazon S3 Transfer Acceleration adalah fitur yang memungkinkan transfer data ke bucket dengan cepat. Transfer Acceleration dikonfigurasi pada level bucket individual. Untuk informasi lebih lanjut

tentang Transfer Acceleration, lihat Mengonfigurasi transfer file yang cepat dan aman menggunakan Amazon S3 Transfer Acceleration.

Multi-Region Access Points menggunakan mekanisme transfer akselerasi serupa sebagai Transfer Acceleration untuk mengirim objek besar melaluiAWS jaringan. Karena itu, Anda tidak perlu menggunakan Transfer Acceleration saat mengirim permintaan melalui Multi-Region Access Point. Peningkatan kinerja transfer ini secara otomatis dimasukkan ke dalam Titik Akses Multi-Wilayah.

Topik

- Izin
- Pembatasan dan batasan Titik Akses Multi-Wilayah
- Perutean permintaan Titik Akses Multi-Wilayah
- Kontrol failover Titik Akses Multi-Wilayah Amazon S3
- Mengkonfigurasi replikasi untuk digunakan dengan Titik Akses Multi-Wilayah
- Menggunakan Titik Akses Multi-Wilayah dengan operasi API yang didukung
- Memantau dan mencatat permintaan yang dilakukan melalui Titik Akses Multi-Wilayah ke sumber daya yang mendasarinya

Izin

Titik Akses Multi-Wilayah Amazon S3 dapat menyederhanakan akses data untuk bucket Amazon S3 dalam beberapaWilayah AWS. Titik Akses Multi-Wilayah diberi nama titik akhir global yang dapat Anda gunakan untuk melakukan operasi objek akses data Amazon S3, sepertiGetObjectdanPutObject. Setiap Titik Akses Multi-Wilayah dapat memiliki izin dan kontrol jaringan yang berbeda untuk setiap permintaan yang dibuat melalui titik akhir global.

Setiap Titik Akses Multi-Wilayah juga dapat menerapkan kebijakan akses khusus yang bekerja bersama dengan kebijakan bucket yang dilampirkan ke bucket yang mendasarinya. Agar permintaan berhasil, semua hal berikut harus mengizinkan operasi:

- Kebijakan Multi-Region Access Point
- Yang mendasariAWS Identity and Access Management(IAM) kebijakan
- Kebijakan bucket yang mendasari (tempat permintaan dirutekan)

Anda dapat mengonfigurasi kebijakan Titik Akses Multi-Wilayah untuk menerima permintaan hanya dari pengguna atau grup IAM tertentu. Untuk contoh bagaimana melakukan ini, lihat Contoh 2

dithe section called “Contoh kebijakan Multi-Region Access Point”. Untuk membatasi akses data Amazon S3 ke jaringan pribadi, Anda dapat mengonfigurasi kebijakan Titik Akses Multi-Wilayah untuk menerima permintaan hanya dari cloud pribadi virtual (VPC).

Misalnya, anggaplah Anda membuat`GetObject`permintaan melalui Multi-Region Access Point dengan menggunakan pengguna yang disebut`AppDataReader`di dalam kamuAWSakun. Untuk membantu memastikan bahwa permintaan tidak akan ditolak,`AppDataReader`pengguna harus diberikan`s3:GetObject`izin oleh Multi-Region Access Point dan oleh setiap bucket yang mendasari Multi-Region Access Point.`AppDataReader`tidak akan dapat mengambil data dari bucket apa pun yang tidak memberikan izin ini.

> **Important**
>
> Mendelegasikan kontrol akses untuk bucket ke kebijakan Titik Akses Multi-Wilayah tidak mengubah perilaku bucket saat bucket diakses langsung melalui nama bucket atau Amazon Resource Name (ARN). Semua operasi yang dilakukan langsung terhadap bucket akan terus bekerja seperti sebelumnya. Pembatasan yang Anda sertakan dalam kebijakan Titik Akses Multi-Wilayah hanya berlaku untuk permintaan yang dibuat melalui Titik Akses Multi-Wilayah tersebut.

Mengelola akses publik ke Titik Akses Multi-Wilayah

Titik Akses Multi-Wilayah mendukung pengaturan Blokir Akses Publik independen untuk setiap Titik Akses Multi-Wilayah. Saat Anda membuat Titik Akses Multi-Wilayah, Anda dapat menentukan pengaturan Blokir Akses Publik yang berlaku untuk Titik Akses Multi-Wilayah tersebut.

> **Note**
>
> Semua pengaturan Blokir Akses Publik yang diaktifkan di bawahMemblokir pengaturan Akses Publik untuk akun ini(di akun Anda sendiri) atauBlokir Pengaturan Publik untuk bucket eksternalmasih berlaku meskipun pengaturan Blokir Akses Publik independen untuk Titik Akses Multi-Wilayah Anda dinonaktifkan.

Untuk setiap permintaan yang dibuat melalui Titik Akses Multi-Wilayah, Amazon S3 mengevaluasi pengaturan Blokir Akses Publik untuk:

- Titik Akses Multi Wilayah

- Bucket yang mendasari (termasuk ember eksternal)
- Akun yang memiliki Titik Akses Multi-Wilayah
- Akun yang memiliki bucket yang mendasarinya (termasuk akun eksternal)

Jika salah satu pengaturan ini menunjukkan bahwa permintaan tersebut harus diblokir, Amazon S3 menolak permintaan tersebut. Untuk informasi selengkapnya tentang fitur Amazon S3 Block Public Access, lihatMelakukan blok akses publik ke penyimpanan Amazon S3 Anda.

> **Important**
>
> Secara default, semua pengaturan Blokir Akses Publik diaktifkan untuk Titik Akses Multi-Wilayah. Anda harus secara eksplisit menonaktifkan pengaturan apa pun yang tidak ingin Anda terapkan ke Titik Akses Multi-Wilayah.
> Anda tidak dapat mengubah pengaturan Blokir Akses Publik untuk Titik Akses Multi-Wilayah setelah dibuat.

Melihat pengaturan Blokir Akses Publik untuk Titik Akses Multi-Wilayah

Untuk melihat pengaturan Blokir Akses Publik untuk Titik Akses Multi-Wilayah

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihTitik Akses Multi Wilayah.
3. Pilih nama Titik Akses Multi-Wilayah yang ingin Anda tinjau.
4. Pilih tab Izin.
5. Di bawahBlokir pengaturan Akses Publik untuk Titik Akses Multi-Wilayah ini, tinjau pengaturan Blokir Akses Publik untuk Titik Akses Multi-Wilayah Anda.

   > **Note**
   >
   > Anda tidak dapat mengedit pengaturan Blokir Akses Publik setelah Titik Akses Multi-Wilayah dibuat. Oleh karena itu, jika Anda akan memblokir akses publik, pastikan bahwa aplikasi Anda bekerja dengan benar tanpa akses publik sebelum Anda membuat Multi-Region Access Point.

Menggunakan kebijakan Titik Akses Multi-Wilayah

Contoh kebijakan Titik Akses Multi-Wilayah berikut memberikan akses pengguna IAM ke daftar dan mengunduh file dari Titik Akses Multi-Wilayah Anda. Untuk menggunakan kebijakan contoh ini, ganti*user input placeholders*dengan informasi Anda sendiri.

```
 {
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::123456789012:user/JohnDoe"
         },
         "Action":[
            "s3:ListBucket",
            "s3:GetObject"
         ],
         "Resource":[
            "arn:aws:s3::111122223333:accesspoint/MultiRegionAccessPoint_alias",
            "arn:aws:s3::111122223333:accesspoint/MultiRegionAccessPoint_alias/object/
*"
         ]
      }
   ]
}
```

Untuk mengaitkan kebijakan Titik Akses Multi-Wilayah Anda dengan Titik Akses Multi-Wilayah yang ditentukan dengan menggunakanAWS Command Line Interface(AWS CLI), gunakan yang berikut`put-multi-region-access-point-policy`perintah. Untuk menggunakan perintah contoh ini, ganti*user input placeholders*dengan informasi Anda sendiri. Setiap Multi-Region Access Point hanya dapat memiliki satu kebijakan, sehingga permintaan yang dibuat untuk`put-multi-region-access-point-policy`tindakan menggantikan kebijakan yang ada yang terkait dengan Titik Akses Multi-Region yang ditentukan.

AWS CLI

```
aws s3control put-multi-region-access-point-policy
--account-id 111122223333
--details { "Name": "DOC-EXAMPLE-BUCKET-MultiRegionAccessPoint",
 "Policy": "{ \"Version\": \"2012-10-17\", \"Statement\": { \"Effect\":
```

```
 \"Allow\", \"Principal\": { \"AWS\": \"arn:aws:iam::111122223333:root
\" }, \"Action\": [\"s3:ListBucket\", \"s3:GetObject\"], \"Resource\":
 [ \"arn:aws:s3::111122223333:accesspoint/MultiRegionAccessPoint_alias",
 \"arn:aws:s3::111122223333:accesspoint/MultiRegionAccessPoint_alias/object/*
\" ] } }" }
```

Untuk menanyakan hasil Anda untuk operasi sebelumnya, gunakan perintah berikut:

AWS CLI

```
aws s3control describe-multi-region-access-point-operation
--account-id 111122223333
--request-token-arn requestArn
```

Untuk mengambil kebijakan Multi-Region Access Point, gunakan perintah berikut:

AWS CLI

```
aws s3control get-multi-region-access-point-policy
--account-id 111122223333
--name=DOC-EXAMPLE-BUCKET-MultiRegionAccessPoint
```

Mengedit kebijakan Titik Akses Multi-Wilayah

Kebijakan Titik Akses Multi-Wilayah (ditulis dalam JSON) menyediakan akses penyimpanan ke bucket Amazon S3 yang digunakan dengan Titik Akses Multi-Wilayah ini. Anda dapat mengizinkan atau menolak prinsipal tertentu untuk melakukan berbagai tindakan pada Titik Akses Multi-Wilayah Anda. Ketika permintaan dialihkan ke bucket melalui Titik Akses Multi-Wilayah, kebijakan akses untuk Titik Akses Multi-Wilayah dan bucket berlaku. Kebijakan akses yang lebih ketat selalu diutamakan.

> **Note**
>
> Jika bucket berisi objek yang dimiliki oleh akun lain, kebijakan Multi-Region Access Point tidak berlaku untuk objek yang dimiliki oleh lainnyaAkun AWS.

Setelah Anda menerapkan kebijakan Titik Akses Multi-Wilayah, kebijakan tidak dapat dihapus. Anda dapat mengedit kebijakan atau membuat kebijakan baru yang menimpa kebijakan yang sudah ada.

Cara mengedit kebijakan Titik Akses Multi-Wilayah

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihTitik Akses Multi Wilayah.
3. Pilih nama Titik Akses Multi-Wilayah yang ingin Anda edit kebijakan.
4. Pilih tab Izin.
5. Gulir ke bawah keKebijakan Titik Akses Multi-Wilayahbagian. PilihMengedituntuk memperbarui kebijakan (di JSON).
6. YangEdit kebijakan Titik Akses Multi-Wilayahhalaman muncul. Anda dapat memasukkan kebijakan langsung ke bidang teks, atau Anda dapat memilihTambahkan pernyataanuntuk memilih elemen kebijakan dari daftar dropdown.

> **Note**
>
> Konsol secara otomatis menampilkan Multi-Region Access Point Amazon Resource Name (ARN), yang dapat Anda gunakan dalam kebijakan. Misalnya kebijakan Titik Akses Multi-Wilayah, lihatthe section called “Contoh kebijakan Multi-Region Access Point”.

Contoh kebijakan Multi-Region Access Point

Dukungan Titik Akses Multi-Wilayah Amazon S3AWS Identity and Access Management(IAM) kebijakan sumber daya. Anda dapat menggunakan kebijakan ini untuk mengontrol penggunaan Titik Akses Multi-Wilayah berdasarkan sumber daya, pengguna, atau kondisi lainnya. Agar aplikasi atau pengguna dapat mengakses objek melalui Titik Akses Multi-Wilayah, Titik Akses Multi-Wilayah dan bucket yang mendasarinya harus mengizinkan akses yang sama.

Untuk mengizinkan akses yang sama ke Titik Akses Multi-Wilayah dan bucket yang mendasarinya, lakukan salah satu hal berikut:

- (Direkomendasikan)Untuk menyederhanakan kontrol akses saat menggunakan Titik Akses Multi-Wilayah Amazon S3, delegasikan kontrol akses untuk bucket Amazon S3 ke Titik Akses Multi-Wilayah. Untuk contoh bagaimana melakukan ini, lihat Contoh 1 di bagian ini.
- Tambahkan izin yang sama yang terdapat dalam kebijakan Titik Akses Multi-Wilayah ke kebijakan bucket yang mendasarinya.

> **Important**
>
> Mendelegasikan kontrol akses untuk bucket ke kebijakan Titik Akses Multi-Wilayah tidak mengubah perilaku bucket saat bucket diakses langsung melalui nama bucket atau Amazon Resource Name (ARN). Semua operasi yang dilakukan langsung terhadap bucket akan terus bekerja seperti sebelumnya. Pembatasan yang Anda sertakan dalam kebijakan Titik Akses Multi-Wilayah hanya berlaku untuk permintaan yang dibuat melalui Titik Akses Multi-Wilayah tersebut.

Example 1 — Mendelegasikan akses ke Titik Akses Multi-Wilayah tertentu dalam kebijakan bucket Anda (untuk akun atau lintas akun yang sama)

Kebijakan bucket contoh berikut memberikan akses bucket penuh ke Titik Akses Multi-Wilayah tertentu. Ini berarti bahwa semua akses ke bucket ini dikendalikan oleh kebijakan yang dilampirkan ke Multi-Region Access Point. Kami menyarankan Anda untuk mengonfigurasi bucket dengan cara ini untuk semua kasus penggunaan yang tidak memerlukan akses langsung ke bucket. Anda dapat menggunakan struktur kebijakan bucket ini untuk Titik Akses Multi-Wilayah di akun yang sama atau di akun lain.

```
{
    "Version": "2012-10-17",
    "Statement" : [
    {
        "Effect": "Allow",
        "Principal" : { "AWS": "*" },
        "Action" : "*",
        "Resource" : [ "Bucket ARN", "Bucket ARN/*"],
        "Condition": {
            "StringEquals" : { "s3:DataAccessPointArn" : "MultiRegionAccessPoint_ARN" }
        }
    }]
}
```

> **Note**
>
> Jika ada beberapa Titik Akses Multi-Wilayah yang Anda berikan akses, pastikan untuk mencantumkan setiap Titik Akses Multi-Wilayah.

Example 2 - Memberikan akses akun ke Titik Akses Multi-Wilayah dalam kebijakan Titik Akses Multi-Wilayah Anda

Kebijakan Titik Akses Multi-Wilayah berikut memungkinkan akun*123456789012*izin untuk daftar dan membaca objek yang terkandung dalam Multi-Region Access Point didefinisikan oleh*MultiRegionAccessPoint_ARN*.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
       "Effect": "Allow",
       "Principal": {
          "AWS":"arn:aws:iam::123456789012:user/JohnDoe"
       },
       "Action":[
          "s3:ListBucket",
          "s3:GetObject"
       ],
       "Resource":[
          "MultiRegionAccessPoint_ARN",
          "MultiRegionAccessPoint_ARN/object/*"
       ]
     }
  ]
}
```

Example 3 - Kebijakan Titik Akses Multi-Wilayah yang memungkinkan daftar bucket

Kebijakan Titik Akses Multi-Wilayah berikut memungkinkan akun*123456789012*izin untuk daftar objek yang terkandung dalam Multi-Region Access Point didefinisikan oleh*MultiRegionAccessPoint_ARN*.

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
       "Effect": "Allow",
       "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/JohnDoe"
        },
        "Action": "s3:ListBucket",
```

```
        "Resource": "MultiRegionAccessPoint_ARN"
       }
    ]
}
```

Pembatasan dan batasan Titik Akses Multi-Wilayah

Titik Akses Multi-Wilayah di Amazon S3 memiliki pembatasan dan batasan berikut.

- Nama Titik Akses Multi-Wilayah:
  - Harus unik dalam satu AWS akun
  - Harus dimulai dengan angka atau huruf kecil
  - Panjang nama harus berkisar antara 3 sampai 50 karakter
  - Tidak dapat diawali atau diakhiri dengan tanda hubung (`-`)
  - Tidak boleh berisi garis bawah (_), huruf besar, atau titik (`.`)
  - Tidak dapat diedit setelah dibuat
- Alias Titik Akses Multi-Wilayah dibuat oleh Amazon S3 dan tidak dapat diedit atau digunakan kembali.
- Anda tidak dapat mengakses data melalui Titik Akses Multi-Wilayah dengan menggunakan titik akhir gateway. Namun, Anda dapat mengakses data melalui Titik Akses Multi-Wilayah dengan menggunakan titik akhir antarmuka. Untuk menggunakannya AWS PrivateLink, Anda harus membuat titik akhir VPC. Untuk informasi selengkapnya, lihat Mengonfigurasi Titik Akses Multi-Wilayah untuk digunakan denganAWS PrivateLink.
- Untuk menggunakan Titik Akses Multi-Wilayah dengan Amazon CloudFront, Anda harus mengonfigurasi Titik Akses Multi-Wilayah sebagai jenis `Custom Origin` distribusi. Untuk informasi selengkapnya tentang berbagai jenis asal, lihat Menggunakan berbagai asal dengan CloudFront distribusi. Untuk informasi selengkapnya tentang penggunaan Titik Akses Multi-Wilayah dengan Amazon CloudFront, lihat Membangun aplikasi berbasis kedekatan aktif aktif di beberapa Wilayah di Blog Penyimpanan.AWS
- Persyaratan minimum Titik Akses Multi-Wilayah:
  - Keamanan Lapisan Pengangkutan (TLS) v1.2
  - Tanda Tangan Versi 4 (SigV4A)

    Titik Akses Multi-Wilayah mendukung Tanda Tangan Versi 4A. Versi SigV4 ini memungkinkan permintaan ditandatangani untuk beberapa Wilayah AWS. Fitur ini berguna dalam operasi API yang dapat menghasilkan akses data dari beberapa Wilayah. Saat menggunakan AWS

SDK, Anda memberikan kredensialnya, dan permintaan ke Titik Akses Multi-Wilayah akan menggunakan Signature Version 4A tanpa konfigurasi tambahan. Pastikan untuk memeriksa kompatibilitas SDK AWS Anda dengan algoritma SigV4A. Untuk informasi selengkapnya tentang Sigv4a, lihat Menandatangani permintaan AWS API di. Referensi Umum AWS

> **Note**
>
> Untuk menggunakan Sigv4a dengan kredensil keamanan sementara—misalnya, saat menggunakan peran AWS Identity and Access Management (IAM) — Anda dapat meminta kredensil sementara dari titik akhir Regional (). AWS Security Token Service AWS STS Jika Anda meminta kredensi sementara dari AWS STS titik akhir global (`sts.amazonaws.com`), maka Anda harus terlebih dahulu mengatur kompatibilitas Wilayah token sesi agar titik akhir global valid di semua. Wilayah AWS Untuk informasi selengkapnya, lihat Mengelola AWS STSWilayah AWS dalam Panduan Pengguna IAM.

- Titik Akses Multi-Wilayah tidak mendukung permintaan anonim.
- Batasan Titik Akses Multi-Wilayah:
  - IPv6 tidak didukung.
  - Bucket Amazon S3 di Outposts tidak didukung.
  - Titik Akses Multi-Region mendukung operasi penyalinan menggunakan Titik Akses Multi-Wilayah hanya sebagai tujuan saat menggunakan ARN Titik Akses Multi-Wilayah.
  - Fitur Operasi Batch S3 tidak didukung.
- AWS SDK tertentu tidak didukung. Untuk mengonfirmasi AWS SDK mana yang didukung untuk Titik Akses Multi-Wilayah, lihat Kompatibilitas dengan AWS SDK.
- Kuota layanan untuk Titik Akses Multi-Wilayah adalah sebagai berikut:
  - Maksimal 100 Titik Akses Multi-Wilayah per akun.
  - Jumlah Wilayah maksimal untuk satu Titik Akses Multi-Wilayah adalah 17.
- Setelah membuat Titik Akses Multi-Wilayah, Anda tidak dapat menambahkan, memodifikasi, atau menghapus bucket dari konfigurasi Titik Akses Multi-Wilayah. Untuk mengubah bucket, Anda harus menghapus seluruh Titik Akses Multi-Wilayah dan membuat titik akses baru. Jika bucket lintas akun di Titik Akses Multi-Wilayah Anda dihapus, satu-satunya cara untuk menyambungkan kembali bucket ini adalah dengan membuat ulang bucket, menggunakan nama dan Wilayah yang sama di akun tersebut.
- Bucket yang mendasarinya (di akun yang sama) yang digunakan di Titik Akses Multi-Wilayah hanya dapat dihapus setelah Titik Akses Multi-Wilayah dihapus.

- Semua permintaan bidang kontrol untuk membuat atau mengelola Titik Akses Multi-Wilayah harus dirutekan ke Wilayah `US West (Oregon)`. Untuk permintaan bidang data Titik Akses Multi-Wilayah, tidak perlu menentukan Wilayah secara khusus.
- Untuk bidang kontrol failover Titik Akses Multi-Wilayah, permintaan harus diarahkan ke salah satu dari lima Wilayah yang didukung berikut ini:
  - `US East (N. Virginia)`
  - `US West (Oregon)`
  - `Asia Pacific (Sydney)`
  - `Asia Pacific (Tokyo)`
  - `Europe (Ireland)`
- Titik Akses Multi-Wilayah Anda hanya mendukung bucket dalam hal berikut: Wilayah AWS
  - `US East (N. Virginia)`
  - `US East (Ohio)`
  - `US West (N. California)`
  - `US West (Oregon)`
  - `Asia Pacific (Mumbai)`
  - `Asia Pacific (Osaka)`
  - `Asia Pacific (Seoul)`
  - `Asia Pacific (Singapore)`
  - `Asia Pacific (Sydney)`
  - `Asia Pacific (Tokyo)`
  - `Canada (Central)`
  - `Europe (Frankfurt)`
  - `Europe (Ireland)`
  - `Europe (London)`
  - `Europe (Paris)`
  - `Europe (Stockholm)`
  - `South America (São Paulo)`

Perutean permintaan Titik Akses Multi-Wilayah

Saat Anda membuat permintaan melalui Titik Akses Multi-Wilayah, Amazon S3 menentukan bucket mana yang terkait dengan Titik Akses Multi-Wilayah yang paling dekat dengan Anda. Amazon S3 kemudian mengarahkan permintaan ke bucket tersebut, terlepas dariAWS Wilayah tempat itu berada.

Setelah Titik Akses Multi-Wilayah merutekan permintaan ke bucket jarak dekat, Amazon S3 memproses permintaan seolah-olah Anda membuatnya langsung ke bucket tersebut. Titik Akses Multi-Wilayah tidak mengetahui konten data bucket Amazon S3. Oleh karena itu, bucket yang mendapat permintaan mungkin tidak berisi data yang diminta. Untuk membuat set data yang konsisten di bucket Amazon S3 yang terkait dengan Cross-Region Access Point, Anda dapat mengkonfigurasi S3 Cross-Region Replication (CRR). Kemudian bucket apa pun dapat memenuhi permintaan dengan sukses.

Amazon S3 mengarahkan permintaan Titik Akses Multi-Region sesuai dengan aturan berikut:

- Amazon S3 mengoptimalkan permintaan untuk dipenuhi sesuai dengan kedekatan. Ini melihat bucket yang didukung oleh Multi-Region Access Point dan menyampaikan permintaan ke bucket yang memiliki kedekatan terdekat.
- Jika permintaan menentukan sumber daya yang ada (misalnya,`GetObject`), Amazon S3 tidak mempertimbangkan nama objek saat memenuhi permintaan. Ini berarti bahwa bahkan jika objek ada dalam satu bucket di Multi-Region Access Point, permintaan Anda dapat dialihkan ke bucket yang tidak berisi objek. Situasi ini akan mengakibatkan pesan kesalahan 404 dikembalikan ke klien.

  Untuk menghindari 404 kesalahan, kami menyarankan Anda mengkonfigurasi S3 Cross-Region Replication (CRR) untuk bucket Anda. Replikasi membantu mengatasi potensi masalah saat objek yang Anda inginkan berada di bucket di Titik Akses Multi-Wilayah, tetapi tidak terletak di bucket tertentu yang dirutekan permintaan Anda. Untuk informasi lebih lanjut tentang konfigurasi replikasi, lihatMengkonfigurasi replikasi untuk digunakan dengan Titik Akses Multi-Wilayah.

  Untuk memastikan bahwa permintaan Anda dipenuhi dengan menggunakan objek tertentu yang Anda inginkan, kami juga menyarankan Anda mengaktifkan versi bucket dan menyertakan ID versi dalam permintaan Anda. Pendekatan ini membantu memastikan bahwa Anda memiliki versi yang benar dari objek yang Anda cari. Bucket dengan versioning diaktifkan memungkinkan Anda memulihkan objek dari penimpaan yang tidak disengaja. Untuk informasi selengkapnya, lihat Menggunakan S3 di bucket S3.

- Jika permintaan adalah untuk membuat sumber daya (misalnya,`PutObject` atau`CreateMultipartUpload`), Amazon S3 memenuhi permintaan dengan menggunakan bucket jarak dekat. Misalnya, pertimbangkan perusahaan video yang ingin mendukung unggahan video dari mana saja di dunia. Ketika pengguna membuat`PUT` permintaan ke Multi-Region Access Point, objek dimasukkan ke dalam bucket dengan kedekatan terdekat. Untuk kemudian membuat video yang diunggah tersedia untuk orang lain di seluruh dunia untuk diunduh dengan latensi terendah, Anda dapat menggunakan CRR dengan replikasi dua arah (dua arah). Menggunakan CRR dengan replikasi dua arah membuat konten semua bucket yang terkait dengan Multi-Region Access Point disinkronkan. Untuk informasi lebih lanjut tentang replikasi dengan Multi-Region Access Point, lihat[Mengkonfigurasi replikasi untuk digunakan dengan Titik Akses Multi-Wilayah](#).

Kontrol failover Titik Akses Multi-Wilayah Amazon S3

Dengan kontrol failover Titik Akses Multi-Wilayah Amazon S3, Anda dapat mempertahankan kelangsungan bisnis selama gangguan lalu lintas Regional, sekaligus memberi aplikasi Anda arsitektur Multi-wilayah untuk memenuhi kebutuhan kepatuhan dan redundansi. Jika lalu lintas Regional terganggu, Anda dapat menggunakan kontrol failover Titik Akses Multi-Wilayah untuk memilih yangWilayah AWS berada di belakang Titik Akses Multi-Wilayah Amazon S3 yang akan memproses permintaan akses data dan penyimpanan.

Untuk mendukung failover, Anda dapat mengatur Titik Akses Multi-Wilayah dalam konfigurasi pasif aktif, dengan lalu lintas yang mengalir ke Wilayah aktif selama kondisi normal, dan Wilayah pasif dalam keadaan siaga untuk failover.

Misalnya, untuk melakukan failover ke pilihanWilayah AWS Anda, Anda mengalihkan lalu lintas dari Wilayah utama (aktif) ke Wilayah sekunder (pasif) Anda. Dalam konfigurasi aktif-pasif seperti ini, satu bucket aktif dan menerima lalu lintas, sedangkan bucket lainnya pasif dan tidak menerima lalu lintas. Bucket pasif digunakan untuk pemulihan bencana. Ketika Anda memulai failover, semua lalu lintas (seperti`GET` atau`PUT` permintaan) diarahkan ke bucket dalam keadaan aktif (dalam satu Wilayah) dan jauh dari bucket dalam keadaan pasif (di Wilayah lain).

Jika Anda mengaktifkan Replikasi S3 Lintas Wilayah (CRR) dengan aturan replikasi dua arah, Anda dapat menjaga bucket Anda disinkronisasi selama failover. Selain itu, jika Anda mengaktifkan CRR dalam konfigurasi aktif-aktif, Titik Akses Multi-Wilayah Amazon S3 juga dapat mengambil data dari lokasi bucket dengan jarak terdekat, yang meningkatkan kinerja aplikasi.

Dukungan Wilayah AWS

Dengan kontrol failover Titik Akses Multi-Wilayah Amazon S3, bucket S3 Anda dapat berada di salah satu dari 17 Wilayah di mana Titik Akses Multi-Wilayah didukung. Anda dapat memulai failover di dua Wilayah pada satu waktu.

> **Note**
>
> Meskipun failover dimulai antara hanya dua Wilayah pada satu waktu, Anda dapat memperbarui status perutean secara terpisah untuk beberapa Wilayah secara bersamaan di Titik Akses Multi-Wilayah Anda.

Topik berikut menunjukkan cara menggunakan dan mengelola kontrol failover Titik Akses Multi-Wilayah Amazon S3.

Topik

- Status perutean Titik Akses Multi-Wilayah Amazon S3
- Menggunakan kontrol failover Titik Akses Multi-Wilayah Amazon S3
- Kesalahan kontrol failover Titik Akses Multi-Wilayah Amazon S3

Status perutean Titik Akses Multi-Wilayah Amazon S3

Konfigurasi failover Titik Akses Multi-Wilayah Amazon S3 Anda menentukan status peruteanWilayah AWS yang digunakan dengan Titik Akses Multi-Wilayah. Anda dapat mengonfigurasi Titik Akses Multi-Wilayah Amazon S3 agar berada dalam keadaan aktif aktif atau status pasif aktif.

- Aktif-aktif - Dalam konfigurasi aktif-aktif, semua permintaan secara otomatis dikirim ke kedekatan terdekatWilayah AWS di Titik Akses Multi-Wilayah Anda. Setelah Multi-Region Access Point telah dikonfigurasi untuk berada dalam keadaan aktif-aktif, semua Wilayah dapat menerima lalu lintas. Jika gangguan lalu lintas terjadi dalam konfigurasi aktif-aktif, lalu lintas jaringan akan secara otomatis diarahkan ke salah satu Wilayah aktif.
- Aktif-pasif - Dalam konfigurasi pasif aktif, Wilayah aktif di Titik Akses Multi-Wilayah Anda menerima lalu lintas dan yang pasif tidak. Jika Anda bermaksud menggunakan kontrol failover S3 untuk memulai failover dalam situasi bencana, siapkan Titik Akses Multi-Wilayah Anda dalam konfigurasi pasif aktif saat Anda menguji dan melakukan perencanaan pemulihan bencana.

Menggunakan kontrol failover Titik Akses Multi-Wilayah Amazon S3

Bagian ini menjelaskan cara mengelola dan menggunakan kontrol failover Amazon S3 Lintas Wilayah Anda dengan menggunakanAWS Management Console.

Ada dua kontrol failover di bagian konfigurasi Failover pada halaman detail Titik Akses Multi-Wilayah Anda diAWS Management Console: Edit status perutean dan Failover. Anda dapat menggunakan kontrol ini sebagai berikut:

- Edit status perutean - Anda dapat mengedit status perutean hingga 17 secara manualWilayah AWS dalam satu permintaan untuk Titik Akses Multi-Wilayah Anda dengan memilih Edit status perutean. Anda dapat menggunakan status Edit routing untuk tujuan berikut:
  - Untuk mengatur atau mengedit status perutean satu atau beberapa Wilayah di Titik Akses Multi-Wilayah Anda
  - Untuk membuat konfigurasi failover untuk Titik Akses Multi-Wilayah Anda dengan mengonfigurasi dua Wilayah agar berada dalam keadaan pasif aktif
  - Untuk gagal secara manual atas Wilayah Anda
  - Untuk mengalihkan lalu lintas antar Wilayah secara manual
- Failover - Saat Anda memulai failover dengan memilih Failover, Anda hanya memperbarui status perutean dua Wilayah yang sudah dikonfigurasi untuk berada dalam keadaan pasif aktif. Selama failover yang Anda mulai dengan memilih Failover, status routing antara dua Wilayah secara otomatis diaktifkan.

Mengedit status perutean Wilayah di Titik Akses Multi-Wilayah Anda

Anda dapat memperbarui status perutean hingga 17 secara manualWilayah AWS dalam satu permintaan untuk Titik Akses Multi-Wilayah Anda dengan memilih status perutean Edit di bagian konfigurasi Failover di halaman detail Titik Akses Multi-Wilayah Anda. Namun, ketika Anda memulai failover dengan memilih Failover, Anda hanya memperbarui status routing dari dua Wilayah yang sudah dikonfigurasi untuk berada dalam keadaan aktif-pasif. Selama failover yang Anda mulai dengan memilih Failover, status routing antara dua Wilayah secara otomatis diaktifkan.

Anda dapat menggunakan status Edit routing (seperti yang dijelaskan dalam prosedur berikut) untuk tujuan berikut:

- Untuk mengatur atau mengedit status perutean satu atau beberapa Wilayah di Titik Akses Multi-Wilayah Anda

- Untuk membuat konfigurasi failover untuk Titik Akses Multi-Wilayah Anda dengan mengonfigurasi dua Wilayah agar berada dalam keadaan pasif aktif
- Untuk gagal secara manual atas Wilayah Anda
- Untuk mengalihkan lalu lintas antar Wilayah secara manual

Menggunakan konsol S3

Untuk memperbarui status perutean Wilayah di Titik Akses Multi-Wilayah

1. Masuk keAWS Management Console.
2. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
3. Pada panel navigasi kiri, pilih Titik Akses Multi-Wilayah.
4. Pilih Titik Akses Lintas Wilayah yang ingin Anda perbarui.
5. Pilih tab Replikasi dan failover.
6. Pilih satu atau beberapa Wilayah yang ingin Anda edit status perutean.

   > **Note**
   >
   > Untuk memulai failover, setidaknya satuWilayah AWS harus ditetapkan sebagai Aktif dan satu Wilayah harus ditetapkan sebagai Pasif di Titik Akses Multi-Wilayah Anda.

7. Pilih Edit status perutean.
8. Di kotak dialog yang muncul, pilih Aktif atau Pasif untuk status Perutean untuk setiap Wilayah.

   Status aktif memungkinkan lalu lintas dialihkan ke Wilayah. Keadaan pasif menghentikan lalu lintas apa pun diarahkan ke Wilayah.

   Jika Anda membuat konfigurasi failover untuk Titik Akses Multi-Wilayah atau memulai failover, setidaknya satuWilayah AWS harus ditetapkan sebagai Aktif dan satu Wilayah harus ditetapkan sebagai Pasif di Titik Akses Multi-Wilayah Anda.
9. Pilih Simpan status perutean. Dibutuhkan sekitar 2 menit agar lalu lintas dialihkan.

Setelah mengirimkan status peruteanWilayah AWS untuk Titik Akses Multi-Wilayah, Anda dapat memverifikasi perubahan status perutean. Untuk memverifikasi perubahan ini, buka Amazon CloudWatch di https://console.aws.amazon.com/cloudwatch/ untuk memantau pergeseran lalu lintas

permintaan data Amazon S3 Anda (misalnya,`GET` dan`PUT` permintaan) antara Wilayah aktif dan pasif. Setiap koneksi yang ada tidak akan dihentikan selama failover. Koneksi yang ada akan berlanjut hingga mencapai status sukses atau gagal.

Menggunakan AWS CLI

> **Note**
>
> Anda dapat menjalankan perintahAWS CLI perutean Titik Akses Multi-Wilayah terhadap salah satu dari lima Wilayah ini:
>
> - `ap-southeast-2`
> - `ap-northeast-1`
> - `us-east-1`
> - `us-west-2`
> - `eu-west-1`

Contoh perintah berikut memperbarui konfigurasi rute Multi-Region Access Point Anda saat ini. Untuk memperbarui status aktif atau pasif bucket, tetapkan`TrafficDialPercentage` nilainya`100` untuk aktif dan`0` untuk pasif. Dalam contoh ini,*`DOC-EXAMPLE-BUCKET-1`* diatur ke aktif, dan*`DOC-EXAMPLE-BUCKET-2`* diatur ke pasif. Untuk menggunakan perintah contoh ini, ganti*`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control submit-multi-region-access-point-routes
--region ap-southeast-2
--account-id 111122223333
--mrap MultiRegionAccessPoint_ARN
--route-updates Bucket=DOC-EXAMPLE-BUCKET-1,TrafficDialPercentage=100
                Bucket=DOC-EXAMPLE-BUCKET-2,TrafficDialPercentage=0
```

Perintah contoh berikut mendapatkan konfigurasi perutean Multi-Region Access Point yang diperbarui. Untuk menggunakan perintah contoh ini, ganti*`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control get-multi-region-access-point-routes
--region eu-west-1
--account-id 111122223333
```

```
--mrap MultiRegionAccessPoint_ARN
```

Mem-mutasi failover

Saat Anda memulai failover dengan memilih Failover di bagian konfigurasi Failover pada halaman detail Titik Akses Multi-Wilayah Anda, lalu lintas permintaan Amazon S3 secara otomatis dialihkan ke alternatifWilayah AWS. Proses failover selesai dalam waktu 2 menit.

Anda dapat memulai failover di duaWilayah AWS sekaligus (dari 17 Wilayah tempat Titik Akses Multi-Wilayah didukung). Peristiwa failover kemudian loginAWS CloudTrail. Setelah kegagalan selesai, Anda dapat memantau lalu lintas Amazon S3 dan pembaruan perutean lalu lintas apa pun ke Wilayah aktif baru di Amazon CloudWatch.

> **Important**
>
> Agar semua metadata dan objek tetap sinkron di seluruh bucket selama replikasi data, sebaiknya buat aturan replikasi dua arah dan aktifkan sinkronisasi modifikasi replika sebelum mengonfigurasi kontrol failover Anda.
> Aturan replikasi dua arah membantu memastikan bahwa ketika data ditulis ke bucket Amazon S3 yang lalu lintas gagal, data tersebut kemudian direplikasi kembali ke bucket sumber. Sinkronisasi modifikasi replika membantu memastikan bahwa metadata objek juga disinkronkan antara bucket selama replikasi dua arah.
> Untuk informasi selengkapnya tentang konfigurasi replikasi untuk mendukung failover, lihatthe section called “Replikasi ember”.

Untuk memulai failover antara bucket yang direplikasi

1. Masuk keAWS Management Console.
2. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
3. Pada panel navigasi kiri, pilih Titik Akses Multi-Wilayah.
4. Pilih Multi-Region Access Point yang ingin Anda gunakan untuk memulai failover.
5. Pilih tab Replikasi dan failover.
6. Gulir ke bawah ke bagian konfigurasi Failover dan pilih duaWilayah AWS.

> **Note**
>
> Untuk memulai failover, setidaknya satuWilayah AWS harus ditetapkan sebagai Aktif dan satu Wilayah harus ditetapkan sebagai Pasif di Titik Akses Multi-Wilayah Anda. Keadaan aktif memungkinkan lalu lintas diarahkan ke Wilayah. Keadaan pasif menghentikan lalu lintas apa pun diarahkan ke Wilayah.

7. Pilih Failover.
8. Pada kotak dialog, pilih failover lagi untuk memulai proses failover. Selama proses ini, status perutean kedua Wilayah secara otomatis diaktifkan. Semua lalu lintas baru diarahkan ke Wilayah yang menjadi aktif, dan lalu lintas berhenti diarahkan ke Wilayah yang menjadi pasif. Dibutuhkan sekitar 2 menit agar lalu lintas dialihkan.

   Setelah Anda memulai proses failover, Anda dapat memverifikasi perubahan lalu lintas Anda. Untuk memverifikasi perubahan ini, buka Amazon CloudWatch di https://console.aws.amazon.com/cloudwatch/ untuk memantau pergeseran lalu lintas permintaan data Amazon S3 Anda (misalnya,`GET` dan`PUT` permintaan) antara Wilayah aktif dan pasif. Setiap koneksi yang ada tidak akan dihentikan selama failover. Koneksi yang ada akan berlanjut hingga mencapai status sukses atau gagal.

Melihat kontrol perutean Titik Akses Multi-Wilayah Amazon S3 Anda

Menggunakan konsol S3

Untuk melihat kontrol perutean untuk Titik Akses Multi-Wilayah Amazon S3 Anda

1. Masuk keAWS Management Console.
2. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
3. Pada panel navigasi kiri, pilih Titik Akses Multi-Wilayah.
4. Pilih Titik Akses Lintas Wilayah yang ingin Anda tinjau.
5. Pilih tab Replikasi dan failover. Halaman ini menampilkan detail dan ringkasan konfigurasi perutean untuk Titik Akses Multi-Wilayah, aturan replikasi terkait, dan metrik replikasi. Anda dapat melihat status routing Wilayah Anda di bagian konfigurasi Failover.

Menggunakan AWS CLI

ContohAWS CLI perintah berikut mendapatkan konfigurasi rute Multi-Region Access Point Anda saat ini untuk Wilayah yang ditentukan. Untuk menggunakan perintah contoh ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
aws s3control get-multi-region-access-point-routes
--region eu-west-1
--account-id 111122223333
--mrap MultiRegionAccessPoint_ARN
```

> **Note**
>
> Perintah ini hanya dapat dijalankan terhadap lima Wilayah ini:
>
> - `ap-southeast-2`
> - `ap-northeast-1`
> - `us-east-1`
> - `us-west-2`
> - `eu-west-1`

Kesalahan kontrol failover Titik Akses Multi-Wilayah Amazon S3

Saat Anda memperbarui konfigurasi failover untuk Titik Akses Multi-Wilayah, Anda mungkin mengalami salah satu kesalahan berikut:

- Permintaan Buruk HTTP 400: Kesalahan ini dapat terjadi jika Anda memasukkan ARN Titik Akses Multi-Wilayah yang tidak valid saat memperbarui konfigurasi failover Anda. Anda dapat mengonfirmasi ARN Titik Akses Multi-Wilayah Anda dengan meninjau kebijakan Titik Akses Multi-Wilayah Anda. Untuk meninjau atau memperbarui kebijakan Titik Akses Multi-Wilayah, lihat Mengedit kebijakan Titik Akses Multi-Wilayah. Kesalahan ini juga dapat terjadi jika Anda menggunakan string kosong atau string acak saat memperbarui kontrol failover Titik Akses Multi-Wilayah Amazon S3 Anda. Pastikan untuk menggunakan format ARN Multi-Wilayah:

  `arn:aws:s3::account-id:accesspoint/MultiRegionAccessPoint_alias`

- HTTP 503 Memperlambat: Kesalahan ini terjadi jika Anda mengirim terlalu banyak permintaan dalam waktu singkat. Permintaan yang ditolak akan menghasilkan kesalahan.

- HTTP 409 Konflik: Kesalahan ini terjadi ketika dua atau lebih permintaan pembaruan konfigurasi rute bersamaan menargetkan satu Titik Akses Multi-Wilayah. Permintaan pertama berhasil, tetapi permintaan lain gagal dengan kesalahan.
- Metode HTTP 405 Tidak Diizinkan: Kesalahan ini terjadi saat Anda memilih Titik Akses Multi-Wilayah dengan hanya satuWilayah AWS saat memulai failover. Anda harus memilih dua Wilayah sebelum Anda dapat memulai failover. Jika tidak, muncul kesalahan.

Mengkonfigurasi replikasi untuk digunakan dengan Titik Akses Multi-Wilayah

Saat Anda mengajukan permintaan ke titik akhir Titik Akses Multi-Wilayah, Amazon S3 secara otomatis merutekan permintaan ke bucket yang paling dekat dengan Anda. Amazon S3 tidak mempertimbangkan konten permintaan saat membuat keputusan ini. Jika Anda membuat permintaan ke `GET` objek, permintaan Anda mungkin dirutekan ke bucket yang tidak memiliki salinan objek ini. Jika itu terjadi, Anda menerima kesalahan kode status HTTP 404 (Tidak Ditemukan). Untuk informasi selengkapnya tentang perutean permintaan Titik Akses Multi-Wilayah, lihat. the section called “Meminta perutean”

Jika Anda ingin Titik Akses Multi-Wilayah dapat mengambil objek terlepas dari bucket mana yang menerima permintaan, Anda harus mengonfigurasi Amazon S3 Cross-Region Replication (CRR).

Misalnya, pertimbangkan Titik Akses Multi-Wilayah dengan tiga bucket:

- Sebuah bucket bernama `my-bucket-usw2` di Region `us-west-2` yang berisi objek `my-image.jpg`
- Sebuah bucket bernama `my-bucket-aps1` di Region `ap-south-1` yang berisi objek `my-image.jpg`
- Bucket bernama `my-bucket-euc1` di Region `eu-central-1` yang tidak berisi objek `my-image.jpg`

Dalam situasi ini, jika Anda membuat `GetObject` permintaan untuk objek`my-image.jpg`, keberhasilan permintaan itu tergantung pada bucket mana yang menerima permintaan Anda. Karena Amazon S3 tidak mempertimbangkan konten permintaan, Amazon S3 dapat merutekan `GetObject` permintaan Anda ke `my-bucket-euc1` bucket jika bucket tersebut merespons kedekatan terdekat. Meskipun objek Anda berada di bucket di Titik Akses Multi-Wilayah, Anda akan mendapatkan

kesalahan HTTP 404 Not Found karena bucket individual yang menerima permintaan Anda tidak memiliki objek tersebut.

Mengaktifkan Replikasi Lintas Wilayah (CRR) membantu menghindari hasil ini. Dengan aturan replikasi yang sesuai, `my-image.jpg` objek disalin ke ember. `my-bucket-euc1` Oleh karena itu, jika Amazon S3 merutekan permintaan Anda ke bucket itu, Anda sekarang dapat mengambil objek tersebut.

Replikasi berfungsi seperti biasa dengan bucket yang ditetapkan ke Titik Akses Multi-Wilayah. Amazon S3 tidak melakukan penanganan replikasi khusus dengan bucket yang ada di Titik Akses Multi-Wilayah. Untuk informasi selengkapnya tentang mengonfigurasi replikasi di bucket Anda, lihat. Menyiapkan replikasi langsung

Rekomendasi untuk menggunakan replikasi dengan Titik Akses Multi-Wilayah

Untuk kinerja replikasi terbaik saat bekerja dengan Titik Akses Multi-Wilayah, kami merekomendasikan hal berikut:

- Konfigurasikan Kontrol Waktu Replikasi S3 (S3 RTC). Untuk mereplikasi data Anda di berbagai Wilayah dalam jangka waktu yang dapat diprediksi, Anda dapat menggunakan S3 RTC. S3 RTC mereplikasi 99,99 persen objek baru yang disimpan di Amazon S3 dalam waktu 15 menit (didukung oleh perjanjian tingkat layanan). Untuk informasi selengkapnya, lihat the section called “Menggunakan Kontrol Waktu Replikasi S3”. Ada biaya tambahan untuk S3 RTC. Untuk selengkapnya, lihat harga Amazon S3.
- Gunakan replikasi dua arah (dua arah) untuk mendukung agar bucket tetap disinkronkan saat bucket diperbarui melalui Titik Akses Multi-Wilayah. Untuk informasi selengkapnya, lihat the section called “Buat aturan replikasi dua arah untuk Titik Akses Multi-Wilayah Anda”.
- Buat Poin Akses Multi-Wilayah lintas akun untuk mereplikasi data ke bucket secara terpisah. Akun AWS Pendekatan ini menyediakan pemisahan tingkat akun, sehingga data dapat diakses dari dan direplikasi di berbagai akun di Wilayah yang berbeda selain bucket sumber. Menyiapkan Poin Akses Multi-Wilayah lintas akun tidak dikenakan biaya tambahan. Jika Anda pemilik bucket tetapi tidak memiliki Titik Akses Multi-Wilayah, Anda hanya membayar untuk transfer data dan biaya permintaan. Pemilik Titik Akses Multi-Wilayah membayar biaya perutean data dan akselerasi internet. Untuk informasi selengkapnya, lihat Harga Amazon S3.
- Aktifkan sinkronisasi modifikasi replika untuk setiap aturan replikasi untuk juga menjaga perubahan metadata ke objek Anda tetap sinkron. Untuk informasi selengkapnya, lihat Mengaktifkan sinkronisasi modifikasi replika.

- Aktifkan CloudWatch metrik Amazon untuk memantau peristiwa replikasi. CloudWatch biaya metrik berlaku. Untuk informasi lebih lanjut, lihat harga Amazon CloudWatch.

Topik

- Buat aturan replikasi satu arah untuk Titik Akses Multi-Wilayah Anda
- Buat aturan replikasi dua arah untuk Titik Akses Multi-Wilayah Anda
- Lihat aturan replikasi untuk Titik Akses Multi-Wilayah Anda

Buat aturan replikasi satu arah untuk Titik Akses Multi-Wilayah Anda

Aturan replikasi memungkinkan penyalinan objek secara otomatis dan asinkron di seluruh ember. Aturan replikasi satu arah membantu memastikan bahwa data sepenuhnya direplikasi dari bucket sumber di bucket satu Wilayah AWS ke tujuan di Wilayah lain. Saat replikasi satu arah disiapkan, aturan replikasi dari bucket sumber (DOC-EXAMPLE-BUCKET-1) ke bucket tujuan (DOC-EXAMPLE-BUCKET-2) dibuat. Seperti semua aturan replikasi, Anda dapat menerapkan aturan replikasi satu arah ke seluruh bucket Amazon S3 atau ke subset objek yang difilter oleh awalan atau tag objek.

> **Important**
>
> Sebaiknya gunakan replikasi satu arah jika pengguna Anda hanya akan mengkonsumsi objek di bucket tujuan Anda. Jika pengguna Anda akan mengunggah atau memodifikasi objek di bucket tujuan Anda, gunakan replikasi dua arah untuk menjaga semua bucket Anda tetap sinkron. Kami juga merekomendasikan replikasi dua arah jika Anda berencana menggunakan Titik Akses Multi-Wilayah untuk failover. Untuk mengatur replikasi dua arah, lihat. the section called “Buat aturan replikasi dua arah untuk Titik Akses Multi-Wilayah Anda”

Untuk membuat aturan replikasi satu arah untuk Titik Akses Multi-Wilayah Anda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Titik Akses Multi-Wilayah.
3. Pilih nama Titik Akses Multi-Wilayah Anda.
4. Pilih tab Replikasi dan failover.
5. Gulir ke bawah ke bagian Aturan replikasi, lalu pilih Buat aturan replikasi. Pastikan Anda memiliki izin yang cukup untuk membuat aturan replikasi, atau pembuatan versi akan dinonaktifkan.

> **Note**
>
> Anda dapat membuat aturan replikasi hanya untuk bucket di akun Anda sendiri. Untuk membuat aturan replikasi untuk bucket eksternal, pemilik bucket harus membuat aturan replikasi untuk bucket tersebut.

6. Pada halaman Buat aturan replikasi, pilih Replikasi objek dari satu atau beberapa bucket sumber ke satu atau beberapa templat bucket tujuan.

> **Important**
>
> Saat Anda membuat aturan replikasi dengan menggunakan templat ini, aturan tersebut menggantikan aturan replikasi yang ada yang sudah ditetapkan ke bucket.
> Untuk menambah atau memodifikasi aturan replikasi yang ada alih-alih menggantinya, buka tab Manajemen setiap bucket di konsol, lalu edit aturan di bagian Aturan replikasi. Anda juga dapat menambah atau memodifikasi aturan replikasi yang ada dengan menggunakanAWS CLI, SDK, atau REST API. Untuk informasi selengkapnya, lihat Konfigurasi Replikasi.

7. Di bagian Sumber dan tujuan, di bawah Bucket sumber, pilih satu atau beberapa bucket yang ingin Anda replikasi objek. Semua bucket (sumber dan tujuan) yang dipilih untuk replikasi harus mengaktifkan Versi S3, dan setiap bucket harus berada di tempat yang berbeda. Wilayah AWS Untuk informasi selengkapnya tentang Pembuatan Versi S3, lihat Menggunakan pembuatan versi di bucket Amazon S3.

   Di bawah Bucket tujuan, pilih satu atau beberapa bucket yang ingin Anda replikasi objek.

> **Note**
>
> Pastikan Anda memiliki izin baca dan replikasi yang diperlukan untuk membuat replikasi, atau Anda akan menemukan kesalahan. Untuk informasi selengkapnya, lihat Membuat peran IAM.

8. Di bagian konfigurasi aturan replikasi, pilih apakah aturan replikasi akan Diaktifkan atau Dinonaktifkan saat dibuat.

> **Note**
>
> Anda tidak dapat memasukkan nama di kotak Nama aturan replikasi. Nama aturan replikasi dihasilkan berdasarkan konfigurasi Anda saat Anda membuat aturan replikasi.

9. Di bagian Lingkup, pilih lingkup yang sesuai untuk replikasi Anda.

   - Untuk mereplikasi seluruh ember, pilih Terapkan ke semua objek di ember.
   - Untuk mereplikasi subset objek dalam bucket, pilih Batasi cakupan aturan ini menggunakan satu atau beberapa filter.

     Anda dapat memfilter objek Anda dengan menggunakan awalan, tag objek, atau kombinasi keduanya.

     - Untuk membatasi replikasi ke semua objek yang memiliki nama yang dimulai dengan string yang sama (misalnya`pictures`), masukkan awalan di kotak Awalan.

       Jika Anda memasukkan awalan yang merupakan nama folder, Anda harus menggunakan pembatas seperti `/` (garis miring ke depan) untuk menunjukkan tingkat hierarki (misalnya,). `pictures/` Untuk informasi selengkapnya tentang awalan, lihat Mengatur objek menggunakan awalan.
     - Untuk mereplikasi semua objek yang memiliki satu atau beberapa tag objek, pilih Tambah tag dan masukkan pasangan kunci-nilai di kotak. Untuk menambahkan tag lain, ulangi prosedur. Untuk informasi lebih lanjut tentang pemberian tanda objek, lihat Mengategorikan penyimpanan Anda menggunakan tag.

10. Gulir ke bawah ke bagian Opsi replikasi tambahan, dan pilih opsi replikasi yang ingin Anda terapkan.

> **Note**
>
> Kami menyarankan Anda menerapkan opsi berikut:
>
> - Kontrol waktu replikasi (RTC) - Untuk mereplikasi data Anda di berbagai Wilayah dalam kerangka waktu yang dapat diprediksi, Anda dapat menggunakan Kontrol Waktu Replikasi S3 (S3 RTC). S3 RTC mereplikasi 99,99 persen objek baru yang disimpan di Amazon S3 dalam waktu 15 menit (didukung oleh perjanjian tingkat layanan). Untuk informasi selengkapnya, lihat the section called “Menggunakan Kontrol Waktu Replikasi S3”.

- Metrik replikasi dan notifikasi — Aktifkan CloudWatch metrik Amazon untuk memantau peristiwa replikasi.
- Hapus replikasi penanda - Hapus penanda yang dibuat oleh operasi penghapusan S3 akan direplikasi. Hapus penanda yang dibuat oleh aturan siklus hidup tidak direplikasi. Untuk informasi selengkapnya, lihat Mereplikasi penanda hapus antar bucket.

Ada biaya tambahan untuk S3 RTC dan metrik CloudWatch replikasi dan notifikasi. Untuk informasi selengkapnya, lihat Harga Amazon S3 dan harga Amazon CloudWatch .

11. Jika Anda menulis aturan replikasi baru yang menggantikan yang sudah ada, pilih Saya mengakui bahwa dengan memilih Buat aturan replikasi, aturan replikasi yang ada ini akan ditimpa.
12. Pilih Buat aturan replikasi untuk membuat dan menyimpan aturan replikasi satu arah baru Anda.

Buat aturan replikasi dua arah untuk Titik Akses Multi-Wilayah Anda

Aturan replikasi memungkinkan penyalinan objek secara otomatis dan asinkron di seluruh ember. Aturan replikasi dua arah (juga dikenal sebagai aturan replikasi dua arah) memastikan bahwa data sepenuhnya disinkronkan antara dua atau lebih bucket yang berbeda. Wilayah AWS Saat replikasi dua arah diatur, aturan replikasi dari bucket sumber (DOC-EXAMPLE-BUCKET-1) ke bucket yang berisi replika (DOC-EXAMPLE-BUCKET-2) dibuat. Kemudian, aturan replikasi kedua dari bucket yang berisi replika (DOC-EXAMPLE-BUCKET-2) ke bucket sumber (DOC-EXAMPLE-BUCKET-1) dibuat.

Seperti semua aturan replikasi, Anda dapat menerapkan aturan replikasi dua arah ke seluruh bucket Amazon S3 atau ke subset objek yang difilter oleh awalan atau tag objek. Anda juga dapat menjaga perubahan metadata ke objek Anda tetap sinkron dengan mengaktifkan sinkronisasi modifikasi replika untuk setiap aturan replikasi. Anda dapat mengaktifkan sinkronisasi modifikasi replika melalui konsol Amazon S3, AWS CLI SDK, AWS Amazon S3 REST API, atau. AWS CloudFormation

Untuk memantau kemajuan replikasi objek dan metadata objek di Amazon CloudWatch, aktifkan metrik dan notifikasi Replikasi S3. Untuk informasi selengkapnya, lihat Memantau kemajuan dengan metrik replikasi dan pemberitahuan peristiwa Amazon S3.

Untuk membuat aturan replikasi dua arah untuk Titik Akses Multi-Wilayah Anda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Titik Akses Multi-Wilayah.
3. Pilih nama Titik Akses Multi-Wilayah yang ingin Anda perbarui.
4. Pilih tab Replikasi dan failover.
5. Gulir ke bawah ke bagian Aturan replikasi, lalu pilih Buat aturan replikasi.
6. Pada halaman Buat aturan replikasi, pilih Replikasi objek di antara semua template bucket yang ditentukan. Objek Replikasi di antara semua templat bucket yang ditentukan menyiapkan replikasi dua arah (dengan kemampuan failover) untuk bucket Anda.

   > **Important**
   >
   > Saat Anda membuat aturan replikasi dengan menggunakan templat ini, aturan tersebut menggantikan aturan replikasi yang ada yang sudah ditetapkan ke bucket.
   > Untuk menambah atau memodifikasi aturan replikasi yang ada alih-alih menggantinya, buka tab Manajemen setiap bucket di konsol, lalu edit aturan di bagian Aturan replikasi. Anda juga dapat menambahkan atau memodifikasi aturan replikasi yang ada dengan menggunakanAWS CLI, AWS SDK, atau Amazon S3 REST API. Untuk informasi selengkapnya, lihat Konfigurasi Replikasi.

7. Di bagian Bucket, pilih setidaknya dua bucket yang ingin Anda replikasi objek. Semua bucket yang dipilih untuk replikasi harus mengaktifkan Versi S3, dan setiap bucket harus berada di tempat yang berbeda. Wilayah AWS Untuk informasi selengkapnya tentang Pembuatan Versi S3, lihat Menggunakan pembuatan versi di bucket Amazon S3.

   > **Note**
   >
   > Pastikan Anda memiliki izin baca dan replikasi yang diperlukan untuk membuat replikasi, atau Anda akan menemukan kesalahan. Untuk informasi selengkapnya, lihat Membuat peran IAM.

8. Di bagian konfigurasi aturan replikasi, pilih apakah aturan replikasi akan Diaktifkan atau Dinonaktifkan saat dibuat.

> **Note**
>
> Anda tidak dapat memasukkan nama di kotak Nama aturan replikasi. Nama aturan replikasi dihasilkan berdasarkan konfigurasi Anda saat Anda membuat aturan replikasi.

9. Di bagian Lingkup, pilih lingkup yang sesuai untuk replikasi Anda.

   - Untuk mereplikasi seluruh ember, pilih Terapkan ke semua objek di ember.
   - Untuk mereplikasi subset objek dalam bucket, pilih Batasi cakupan aturan ini menggunakan satu atau beberapa filter.

     Anda dapat memfilter objek Anda dengan menggunakan awalan, tag objek, atau kombinasi keduanya.

     - Untuk membatasi replikasi ke semua objek yang memiliki nama yang dimulai dengan string yang sama (misalnya`pictures`), masukkan awalan di kotak Awalan.

       Jika Anda memasukkan awalan yang merupakan nama folder, Anda harus menggunakan `/` (garis miring ke depan) sebagai karakter terakhir (misalnya,`pictures/`).
     - Untuk mereplikasi semua objek yang memiliki satu atau beberapa tag objek, pilih Tambah tag dan masukkan pasangan kunci-nilai di kotak. Untuk menambahkan tag lain, ulangi prosedur. Untuk informasi lebih lanjut tentang pemberian tanda objek, lihat Mengategorikan penyimpanan Anda menggunakan tag.
10. Gulir ke bawah ke bagian Opsi replikasi tambahan, dan pilih opsi replikasi yang ingin Anda terapkan.

> **Note**
>
> Kami menyarankan Anda menerapkan opsi berikut, terutama jika Anda bermaksud mengonfigurasi Titik Akses Multi-Wilayah untuk mendukung failover:
>
> - Kontrol waktu replikasi (RTC) - Untuk mereplikasi data Anda di berbagai Wilayah dalam kerangka waktu yang dapat diprediksi, Anda dapat menggunakan Kontrol Waktu Replikasi S3 (S3 RTC). S3 RTC mereplikasi 99,99 persen objek baru yang disimpan di Amazon S3 dalam waktu 15 menit (didukung oleh perjanjian tingkat layanan). Untuk informasi selengkapnya, lihat the section called “Menggunakan Kontrol Waktu Replikasi S3”.

- Metrik replikasi dan notifikasi — Aktifkan CloudWatch metrik Amazon untuk memantau peristiwa replikasi.
- Hapus replikasi penanda - Hapus penanda yang dibuat oleh operasi penghapusan S3 akan direplikasi. Hapus penanda yang dibuat oleh aturan siklus hidup tidak direplikasi. Untuk informasi selengkapnya, lihat Mereplikasi penanda hapus antar bucket.
- Sinkronisasi modifikasi replika — Aktifkan sinkronisasi modifikasi replika untuk setiap aturan replikasi untuk juga menjaga perubahan metadata ke objek Anda tetap sinkron. Untuk informasi selengkapnya, lihat Mengaktifkan sinkronisasi modifikasi replika.

Ada biaya tambahan untuk S3 RTC dan metrik CloudWatch replikasi dan notifikasi. Untuk informasi selengkapnya, lihat Harga Amazon S3 dan harga Amazon CloudWatch.

11. Jika Anda menulis aturan replikasi baru yang menggantikan yang sudah ada, pilih Saya mengakui bahwa dengan memilih Buat aturan replikasi, aturan replikasi yang ada ini akan ditimpa.
12. Pilih Buat aturan replikasi untuk membuat dan menyimpan aturan replikasi dua arah baru Anda.

Lihat aturan replikasi untuk Titik Akses Multi-Wilayah Anda

Dengan Titik Akses Multi-Wilayah, Anda dapat mengatur aturan replikasi satu arah atau aturan replikasi dua arah (dua arah). Untuk informasi tentang cara mengelola aturan replikasi, lihat Mengelola aturan replikasi menggunakan konsol Amazon S3.

Untuk melihat aturan replikasi untuk Titik Akses Multi-Wilayah Anda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Titik Akses Multi-Wilayah.
3. Pilih nama Titik Akses Multi-Wilayah Anda.
4. Pilih tab Replikasi dan failover.
5. Gulir ke bawah ke bagian Aturan replikasi. Bagian ini mencantumkan semua aturan replikasi yang telah dibuat untuk Titik Akses Multi-Wilayah Anda.

> **Note**
>
> Jika Anda telah menambahkan bucket dari akun lain ke Titik Akses Multi-Wilayah ini, Anda harus memiliki `s3:GetBucketReplication` izin dari pemilik bucket untuk melihat aturan replikasi bucket tersebut.

Menggunakan Titik Akses Multi-Wilayah dengan operasi API yang didukung

Amazon S3 menyediakan serangkaian operasi untuk mengelola Titik Akses Multi-Wilayah. Amazon S3 memproses beberapa operasi ini secara sinkron dan beberapa secara asinkron. Saat Anda menginvoke operasi asinkron, Amazon S3 terlebih dahulu akan mengotorisasi operasi yang diminta secara sinkron. Jika otorisasi berhasil, Amazon S3 akan memberikan token yang dapat Anda gunakan untuk melacak kemajuan dan hasil operasi yang diminta.

> **Note**
>
> Permintaan yang dibuat melalui konsol Amazon S3 selalu sinkron. Konsol menunggu hingga permintaan selesai sebelum memperbolehkan Anda mengirimkan permintaan lain.

Anda dapat melihat status saat ini dan hasil operasi asinkron menggunakan konsol, atau Anda dapat menggunakannya `DescribeMultiRegionAccessPointOperation` di, AWS SDK AWS CLI, atau REST API. Amazon S3 memberikan token pelacakan untuk merespons operasi asinkron. Anda menyertakan token pelacakan itu sebagai argumen untuk `DescribeMultiRegionAccessPointOperation`. Saat Anda menyertakan token pelacakan, Amazon S3 kemudian memberikan status saat ini dan hasil operasi yang ditentukan, termasuk kesalahan atau informasi sumber daya yang relevan. Amazon S3 melakukan operasi `DescribeMultiRegionAccessPointOperation` secara sinkron.

Semua permintaan bidang kontrol untuk membuat atau mengelola Titik Akses Multi-Wilayah harus dirutekan ke Wilayah `US West (Oregon)`. Untuk permintaan bidang data Titik Akses Multi-Wilayah, tidak perlu menentukan Wilayah secara khusus. Untuk bidang kontrol failover Titik Akses Multi-Wilayah, permintaan harus dirutekan ke salah satu dari lima Wilayah yang didukung. Untuk informasi selengkapnya tentang Wilayah yang didukung Titik Akses Multi-Wilayah, lihatPembatasan dan batasan Titik Akses Multi-Wilayah.

Selain itu, Anda harus memberikan `s3:ListAllMyBuckets` izin kepada pengguna, peran, atau entitas lain AWS Identity and Access Management (IAM) yang membuat permintaan untuk mengelola Titik Akses Multi-Wilayah.

Berikut ini adalah contoh cara menggunakan Titik Akses Multi-Wilayah dengan operasi yang kompatibel di Amazon S3.

Topik

- Kompatibilitas Titik Akses Multi-Wilayah dengan Layanan AWS dan AWS SDK
- Kompatibilitas Titik Akses Multi-Wilayah dengan operasi S3
- Lihat konfigurasi perutean Titik Akses Multi-Wilayah
- Perbarui kebijakan bucket Amazon S3
- Memperbarui konfigurasi rute Titik Akses Multi-Wilayah
- Tambahkan objek ke bucket di Titik Akses Multi-Wilayah
- Mengambil objek dari Titik Akses Multi-Wilayah
- Buat daftar objek yang disimpan dalam bucket yang mendasari Titik Akses Multi-Wilayah Anda
- Gunakan URL yang telah ditetapkan sebelumnya dengan Titik Akses Multi-Wilayah
- Gunakan bucket yang dikonfigurasi dengan Pembayaran Pemohon dengan Titik Akses Multi-Wilayah

Kompatibilitas Titik Akses Multi-Wilayah dengan Layanan AWS dan AWS SDK

Untuk menggunakan Titik Akses Multi-Wilayah dengan aplikasi yang memerlukan nama bucket Amazon S3, gunakan Nama Sumber Daya Amazon (ARN) dari Titik Akses Multi-Wilayah saat membuat permintaan menggunakan SDK. AWS Untuk memeriksa AWS SDK mana yang kompatibel dengan Titik Akses Multi-Wilayah, lihat Kompatibilitas dengan AWS SDK.

Kompatibilitas Titik Akses Multi-Wilayah dengan operasi S3

Anda dapat menggunakan operasi API bidang data Amazon S3 berikut untuk melakukan tindakan pada objek dalam bucket yang terkait dengan Titik Akses Multi-Wilayah Anda. Operasi S3 berikut dapat menerima ARN Titik Akses Multi-Wilayah:

- AbortMultipartUpload
- CompleteMultipartUpload

- CreateMultipartUpload
- DeleteObject
- DeleteObjectTagging
- GetObject
- GetObjectAcl
- GetObjectLegalHold
- GetObjectRetention
- GetObjectTagging
- HeadObject
- ListMultipartUploads
- ListObjectsV2
- ListParts
- PutObject
- PutObjectAcl
- PutObjectLegalHold
- PutObjectRetention
- PutObjectTagging
- RestoreObject
- UploadPart

> **Note**
>
> Titik Akses Multi-Region mendukung operasi penyalinan menggunakan Titik Akses Multi-Wilayah hanya sebagai tujuan saat menggunakan ARN Titik Akses Multi-Wilayah.

Anda dapat menggunakan operasi bidang kontrol Amazon S3 berikut untuk membuat dan mengelola Titik Akses Multi-Wilayah:

- CreateMultiRegionAccessPoint
- DescribeMultiRegionAccessPointOperation

- GetMultiRegionAccessPoint
- GetMultiRegionAccessPointPolicy
- GetMultiRegionAccessPointPolicyStatus
- GetMultiRegionAccessPointRoutes
- ListMultiRegionAccessPoints
- PutMultiRegionAccessPointPolicy
- SubmitMultiRegionAccessPointRoutes

Lihat konfigurasi perutean Titik Akses Multi-Wilayah

AWS CLI

Contoh perintah berikut mengambil konfigurasi rute Titik Akses Multi-Wilayah sehingga Anda dapat melihat status perutean saat ini untuk bucket Anda. Untuk menggunakan contoh perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control get-multi-region-access-point-routes
--region eu-west-1
--account-id 111122223333
--mrap arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap
```

SDK for Java

Kode SDK for Java berikut mengambil konfigurasi rute Titik Akses Multi-Wilayah sehingga Anda dapat melihat status perutean saat ini untuk bucket Anda. Untuk menggunakan contoh sintaks ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
S3ControlClient s3ControlClient = S3ControlClient.builder()
    .region(Region.US_EAST_1)
    .credentialsProvider(credentialsProvider)
    .build();

GetMultiRegionAccessPointRoutesRequest request =
 GetMultiRegionAccessPointRoutesRequest.builder()
    .accountId("111122223333")
    .mrap("arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap")
    .build();
```

```
GetMultiRegionAccessPointRoutesResponse response =
 s3ControlClient.getMultiRegionAccessPointRoutes(request);
```

SDK for JavaScript

SDK berikut untuk JavaScript kode mengambil konfigurasi rute Titik Akses Multi-Wilayah sehingga Anda dapat melihat status perutean saat ini untuk bucket Anda. Untuk menggunakan contoh sintaks ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
const REGION = 'us-east-1'

const s3ControlClient = new S3ControlClient({
  region: REGION
})

export const run = async () => {
  try {
    const data = await s3ControlClient.send(
      new GetMultiRegionAccessPointRoutesCommand({
        AccountId: '111122223333',
        Mrap: 'arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap',
      })
    )
    console.log('Success', data)
    return data
  } catch (err) {
    console.log('Error', err)
  }
}

run()
```

SDK for Python

SDK berikut untuk kode Python mengambil konfigurasi rute Titik Akses Multi-Wilayah sehingga Anda dapat melihat status perutean saat ini untuk bucket Anda. Untuk menggunakan contoh sintaks ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
s3.get_multi_region_access_point_routes(
        AccountId=111122223333,
        Mrap=arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap)['Routes']
```

Perbarui kebijakan bucket Amazon S3

Untuk memberikan akses yang tepat, Anda juga harus memperbarui kebijakan bucket Amazon S3 yang mendasarinya. Contoh berikut mendelegasikan kontrol akses ke kebijakan Titik Akses Multi-Wilayah. Setelah Anda mendelegasikan kontrol akses ke kebijakan Titik Akses Multi-Wilayah, kebijakan bucket tidak lagi digunakan untuk kontrol akses saat permintaan dibuat melalui Titik Akses Multi-Wilayah.

Berikut adalah contoh kebijakan bucket yang mendelegasikan kontrol akses ke kebijakan Titik Akses Multi-Wilayah. Untuk menggunakan contoh kebijakan bucket ini, ganti *`user input placeholders`* dengan informasi Anda sendiri. Untuk menerapkan kebijakan ini melalui AWS CLI `put-bucket-policy` perintah, seperti yang ditunjukkan pada contoh berikutnya, simpan kebijakan dalam file, misalnya,`policy.json`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": { "AWS": "*" },
    "Effect": "Allow",
    "Action": ["s3:*"],
    "Resource": ["arn:aws:s3:::111122223333/*", "arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
    "Condition": {
      "StringEquals": {
        "s3:DataAccessPointAccount": "444455556666"
      }
    }
  }
}
```

Contoh perintah `put-bucket-policy` berikut mengaitkan kebijakan bucket S3 yang diperbarui dengan bucket S3 Anda:

```
aws s3api put-bucket-policy
  --bucket DOC-EXAMPLE-BUCKET
  --policy file:///tmp/policy.json
```

Memperbarui konfigurasi rute Titik Akses Multi-Wilayah

Contoh perintah berikut memperbarui konfigurasi rute Titik Akses Multi-Wilayah. Perintah rute Titik Akses Multi-Wilayah dapat dijalankan terhadap lima Wilayah berikut:

- `ap-southeast-2`
- `ap-northeast-1`
- `us-east-1`
- `us-west-2`
- `eu-west-1`

Dalam konfigurasi perutean Titik Akses Multi-Wilayah, Anda dapat mengatur bucket ke status perutean aktif atau pasif. Bucket aktif menerima lalu lintas, sedangkan bucket pasif tidak. Anda dapat menyetel status perutean bucket dengan mengatur nilai `TrafficDialPercentage` bucket ke `100` untuk aktif atau `0` untuk pasif.

AWS CLI

Contoh perintah berikut ini memperbarui konfigurasi perutean Titik Akses Multi-Wilayah Anda. Dalam contoh ini, *`DOC-EXAMPLE-BUCKET1`* diatur ke status aktif dan *`DOC-EXAMPLE-BUCKET2`* diatur ke pasif. Untuk menggunakan contoh perintah ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control submit-multi-region-access-point-routes
--region ap-southeast-2
--account-id 111122223333
--mrap arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap
--route-updates Bucket=DOC-EXAMPLE-BUCKET1,TrafficDialPercentage=100
                Bucket=DOC-EXAMPLE-BUCKET2,TrafficDialPercentage=0
```

SDK for Java

Kode SDK for Java berikut memperbarui konfigurasi perutean Titik Akses Multi-Wilayah Anda. Untuk menggunakan contoh sintaks ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
S3ControlClient s3ControlClient = S3ControlClient.builder()
    .region(Region.ap-southeast-2)
    .credentialsProvider(credentialsProvider)
    .build();

SubmitMultiRegionAccessPointRoutesRequest request =
 SubmitMultiRegionAccessPointRoutesRequest.builder()
    .accountId("111122223333")
```

```
    .mrap("arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap")
    .routeUpdates(
        MultiRegionAccessPointRoute.builder()
            .region("eu-west-1")
            .trafficDialPercentage(100)
            .build(),
        MultiRegionAccessPointRoute.builder()
            .region("ca-central-1")
            .bucket("111122223333")
            .trafficDialPercentage(0)
            .build()
    )
    .build();

SubmitMultiRegionAccessPointRoutesResponse response =
 s3ControlClient.submitMultiRegionAccessPointRoutes(request);
```

SDK for JavaScript

SDK berikut untuk JavaScript kode memperbarui konfigurasi rute Titik Akses Multi-Wilayah Anda. Untuk menggunakan contoh sintaks ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
const REGION = 'ap-southeast-2'

const s3ControlClient = new S3ControlClient({
  region: REGION
})

export const run = async () => {
  try {
    const data = await s3ControlClient.send(
      new SubmitMultiRegionAccessPointRoutesCommand({
        AccountId: '111122223333',
        Mrap: 'arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap',
        RouteUpdates: [
          {
            Region: 'eu-west-1',
            TrafficDialPercentage: 100,
          },
          {
            Region: 'ca-central-1',
            Bucket: 'DOC-EXAMPLE-BUCKET1',
```

```
            TrafficDialPercentage: 0,
          },
        ],
      })
    )
    console.log('Success', data)
    return data
  } catch (err) {
    console.log('Error', err)
  }
}

run()
```

SDK for Python

Kode SDK untuk Python berikut ini memperbarui konfigurasi rute Titik Akses Multi-Wilayah Anda. Untuk menggunakan contoh sintaks ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
s3.submit_multi_region_access_point_routes(
        AccountId=111122223333,
        Mrap=arn:aws:s3::111122223333:accesspoint/abcdef0123456.mrap,
        RouteUpdates= [{
            'Bucket': DOC-EXAMPLE-BUCKET,
            'Region': ap-southeast-2,
            'TrafficDialPercentage': 10
        }])
```

Tambahkan objek ke bucket di Titik Akses Multi-Wilayah

Untuk menambahkan objek ke bucket yang terkait dengan Titik Akses Multi-Wilayah, Anda dapat menggunakan operasi PutObject. Agar semua bucket di Titik Akses Multi-Wilayah tetap sinkron, aktifkan Replikasi Lintas Wilayah.

> **Note**
>
> Untuk menggunakan operasi ini, Anda harus memiliki izin `s3:PutObject` untuk Titik Akses Multi-Wilayah. Untuk informasi selengkapnya tentang persyaratan izin Titik Akses Multi-Wilayah, lihat Izin.

AWS CLI

Contoh permintaan bidang data berikut mengunggah *example.txt* ke Titik Akses Multi-Wilayah yang ditentukan. Untuk menggunakan contoh ini, ganti *user input placeholders*dengan informasi Anda sendiri.

```
aws s3api put-object --bucket
 arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap --key example.txt --
body example.txt
```

SDK for Java

```
S3Client s3Client = S3Client.builder()
        .build();

PutObjectRequest objectRequest = PutObjectRequest.builder()
        .bucket("arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap")
        .key("example.txt")
        .build();

s3Client.putObject(objectRequest, RequestBody.fromString("Hello S3!"));
```

SDK for JavaScript

```
const client = new S3Client({});

async function putObjectExample() {
    const command = new PutObjectCommand({
        Bucket: "arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap",
        Key: "example.txt",
        Body: "Hello S3!",
    });

    try {
        const response = await client.send(command);
        console.log(response);
    } catch (err) {
        console.error(err);
    }
}
```

SDK for Python

```
import boto3

client = boto3.client('s3')
client.put_object(
    Bucket='arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap',
    Key='example.txt',
    Body='Hello S3!'
)
```

Mengambil objek dari Titik Akses Multi-Wilayah

Untuk mengambil objek dari Titik Akses Multi-Wilayah, Anda dapat menggunakan operasi GetObject.

> **Note**
>
> Untuk menggunakan operasi API ini, Anda harus memiliki izin `s3:GetObject` untuk Titik Akses Multi-Wiayah. Untuk informasi selengkapnya tentang persyaratan izin Titik Akses Multi-Wilayah, lihat Izin.

AWS CLI

Contoh permintaan bidang data berikut mengambil *`example.txt`* dari Titik Akses Multi-Wilayah yang ditentukan dan mengunduhnya sebagai *`downloaded_example.txt`*. Untuk menggunakan contoh ini, ganti *`user input placeholders`*dengan informasi Anda sendiri.

```
aws s3api get-object --bucket
 arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap --
key example.txt downloaded_example.txt
```

SDK for Java

```
S3Client s3 = S3Client
   .builder()
   .build();

GetObjectRequest getObjectRequest = GetObjectRequest.builder()
```

```
    .bucket("arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap")
    .key("example.txt")
    .build();

s3Client.getObject(getObjectRequest);
```

SDK for JavaScript

```
const client = new S3Client({})

async function getObjectExample() {
    const command = new GetObjectCommand({
        Bucket: "arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap",
        Key: "example.txt"
    });

    try {
        const response = await client.send(command);
        console.log(response);
    } catch (err) {
        console.error(err);
    }
}
```

SDK for Python

```
import boto3

client = boto3.client('s3')
client.get_object(
    Bucket='arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap',
    Key='example.txt'
)
```

Buat daftar objek yang disimpan dalam bucket yang mendasari Titik Akses Multi-Wilayah Anda

Untuk mengembalikan daftar objek yang disimpan dalam bucket yang mendasari Titik Akses Multi-Wilayah, gunakan operasi ListObjectsV2. Dalam contoh perintah berikut, semua objek untuk Titik Akses Multi-Wilayah yang ditentukan dicantumkan dengan menggunakan ARN untuk Titik Akses Multi-Wilayah. Dalam hal ini, ARN Titik Akses Multi-Wilayah adalah:

`arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap`

> **Note**
>
> Untuk menggunakan operasi API ini, Anda harus memiliki izin `s3:ListBucket` untuk Titik Akses Multi-Wilayah dan bucket yang mendasarinya. Untuk informasi selengkapnya tentang persyaratan izin Titik Akses Multi-Wilayah, lihat Izin.

AWS CLI

Contoh permintaan bidang data berikut mencantumkan objek dalam bucket yang mendasari Titik Akses Multi-Wilayah yang ditentukan oleh ARN. Untuk menggunakan contoh ini, ganti *user input placeholders*dengan informasi Anda sendiri.

```
aws s3api list-objects-v2 --bucket
 arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap
```

SDK for Java

```
S3Client s3Client = S3Client.builder()
        .build();

String bucketName = "arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap";

ListObjectsV2Request listObjectsRequest = ListObjectsV2Request
    .builder()
    .bucket(bucketName)
    .build();

 s3Client.listObjectsV2(listObjectsRequest);
```

SDK for JavaScript

```
const client = new S3Client({});

async function listObjectsExample() {
    const command = new ListObjectsV2Command({
        Bucket: "arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap",
    });
```

```
    try {
        const response = await client.send(command);
        console.log(response);
    } catch (err) {
        console.error(err);
    }
}
```

SDK for Python

```
import boto3

client = boto3.client('s3')
client.list_objects_v2(
    Bucket='arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap'
)
```

Gunakan URL yang telah ditetapkan sebelumnya dengan Titik Akses Multi-Wilayah

Anda dapat menggunakan URL yang telah ditetapkan sebelumnya untuk membuat URL sehingga orang lain dapat mengakses bucket Amazon S3 Anda melalui Titik Akses Multi-Wilayah Amazon S3. Saat membuat URL yang telah ditetapkan sebelumnya, kaitkan dengan tindakan objek tertentu, seperti unggah S3 (`PutObject`) atau unduhan S3 (`GetObject`). Anda dapat membagikan URL yang telah ditetapkan sebelumnya, dan siapa pun yang memiliki akses ke URL dapat melakukan tindakan yang disertakan dalam URL seolah-olah mereka adalah pengguna yang membuat URL sendiri.

URL yang telah ditetapkan sebelumnya memiliki masa berlaku. Ketika masa berlaku habis, URL tidak akan berfungsi lagi.

Sebelum Anda menggunakan Titik Akses Multi-Wilayah S3 dengan URL yang telah ditetapkan sebelumnya, periksa AWS kompatibilitas SDK dengan algoritma SigV4A. Verifikasi bahwa versi SDK Anda mendukung Sigv4A sebagai implementasi penandatanganan yang digunakan untuk menandatangani permintaan global Wilayah AWS . Untuk informasi selengkapnya tentang penggunaan URL yang telah ditetapkan sebelumnya dengan Amazon S3, lihat Berbagi objek menggunakan URL yang telah ditetapkan sebelumnya.

Contoh berikut menunjukkan cara menggunakan Titik Akses Multi-Wiayah dengan URL yang telah ditetapkan sebelumnya. Untuk menggunakan contoh ini, ganti *`user input placeholders`*dengan informasi Anda sendiri.

AWS CLI

```
aws s3 presign
 arn:aws:s3::123456789012:accesspoint/MultiRegionAccessPoint_alias/example-file.txt
```

SDK for Python

```
import logging
import boto3
from botocore.exceptions import ClientError

s3_client = boto3.client('s3',aws_access_key_id='xxx',aws_secret_access_key='xxx')
s3_client.generate_presigned_url(HttpMethod='PUT',ClientMethod="put_object",
 Params={'Bucket':'arn:aws:s3::123456789012:accesspoint/
abcdef0123456.mrap','Key':'example-file'})
```

SDK for Java

```
S3Presigner s3Presigner = S3Presigner.builder()
    .credentialsProvider(StsAssumeRoleCredentialsProvider.builder()
        .refreshRequest(assumeRole)
        .stsClient(stsClient)
        .build())
    .build();

GetObjectRequest getObjectRequest = GetObjectRequest.builder()
    .bucket("arn:aws:s3::123456789012:accesspoint/abcdef0123456.mrap")
    .key("example-file")
    .build();

GetObjectPresignRequest preSignedReq = GetObjectPresignRequest.builder()
    .getObjectRequest(getObjectRequest)
    .signatureDuration(Duration.ofMinutes(10))
    .build();

PresignedGetObjectRequest presignedGetObjectRequest =
 s3Presigner.presignGetObject(preSignedReq);
```

> **Note**
>
> Untuk menggunakan Sigv4a dengan kredensi keamanan sementara—misalnya, saat menggunakan peran IAM—pastikan Anda meminta kredensi sementara dari titik akhir Regional di (), bukan titik akhir global. AWS Security Token Service AWS STS Jika Anda menggunakan endpoint global for AWS STS (`sts.amazonaws.com`), AWS STS akan menghasilkan kredensi sementara dari titik akhir global, yang tidak didukung oleh Sig4A. Akibatnya, akan terjadi kesalahan. Untuk mengatasi masalah ini, gunakan salah satu titik akhir Regional yang terdaftar untuk AWS STS.

Gunakan bucket yang dikonfigurasi dengan Pembayaran Pemohon dengan Titik Akses Multi-Wilayah

Jika bucket S3 yang terkait dengan Titik Akses Multi-Wilayah Anda dikonfigurasi untuk menggunakan Pembayaran Pemohon, pemohon akan membayar permintaan bucket, unduhan, dan biaya terkait Titik Akses Multi-Wilayah. Untuk informasi selengkapnya, lihat Harga Amazon S3.

Berikut adalah contoh permintaan bidang data ke Titik Akses Multi-Wilayah yang terhubung ke bucket Pembayaran Pemohon.

AWS CLI

Untuk mengunduh objek dari Titik Akses Multi-Wilayah yang terhubung ke bucket Pembayaran Pemohon, Anda harus menentukan `--request-payer requester` sebagai bagian dari permintaan Anda get-object. Anda juga harus menentukan nama file di dalam bucket dan lokasi penyimpanan file yang diunduh.

```
aws s3api get-object --bucket MultiRegionAccessPoint_ARN --request-payer requester
 --key example-file-in-bucket.txt example-location-of-downloaded-file.txt
```

SDK for Java

Untuk mengunduh objek dari Titik Akses Multi-Wilayah yang terhubung ke bucket Pembayaran Pemohon, Anda harus menentukan `RequestPayer.REQUESTER` sebagai bagian dari permintaan `GetObject` Anda. Anda juga harus menentukan nama file dalam bucket, serta lokasi penyimpanannya.

```
GetObjectResponse getObjectResponse = s3Client.getObject(GetObjectRequest.builder()
```

```
    .key("example-file.txt")
    .bucket("arn:aws:s3::
123456789012:accesspoint/abcdef0123456.mrap")
    .requestPayer(RequestPayer.REQUESTER)
    .build()
).response();
```

Memantau dan mencatat permintaan yang dilakukan melalui Titik Akses Multi-Wilayah ke sumber daya yang mendasarinya

Permintaan log Amazon S3 yang dibuat melalui Titik Akses dan permintaan yang dibuat ke operasi API yang mengelolanya, seperti`CreateMultiRegionAccessPoint` dan`GetMultiRegionAccessPointPolicy`. Permintaan yang dibuat ke Amazon S3 melalui titik akses muncul di log akses server Amazon S3 Anda dengan nama host titik akses Multi-Wilayah.AWS CloudTrail Nama host titik akses mengambil formulir`MRAP_alias.accesspoint.s3-global.amazonaws.com`. Sebagai contoh, asumsikan bahwa Anda memiliki bucket berikut dan konfigurasi titik akses Multi-Wilayah:

- Sebuah bucket bernama`my-bucket-usw2` di Region`us-west-2` yang berisi objek`my-image.jpg`.
- Sebuah bucket bernama`my-bucket-aps1` di Region`ap-south-1` yang berisi objek`my-image.jpg`.
- Bucket bernama`my-bucket-euc1` di Region`eu-central-1` yang tidak berisi objek bernama`my-image.jpg`.
- Titik Akses Multi-Wilayah bernama`my-mrap` dengan alias`mfzwi23gnjvgw.mrap` yang dikonfigurasi untuk memenuhi permintaan dari ketiga bucket.
- IDAWS akun Anda adalah`123456789012`.

Permintaan yang dibuat untuk mengambil`my-image.jpg` langsung melalui salah satu bucket muncul di log Anda dengan nama host dari`bucket_name.s3.Region.amazonaws.com`.

Jika Anda membuat permintaan melalui Titik Akses Multi-Wilayah, Amazon S3 pertama-tama menentukan bucket mana di Wilayah berbeda yang paling dekat. Setelah Amazon S3 menentukan bucket mana yang akan digunakan untuk memenuhi permintaan, ia akan mengirimkan permintaan ke bucket tersebut dan mencatat operasi dengan menggunakan nama host Multi-Region Access Point. Dalam contoh ini, jika Amazon S3 menyampaikan permintaan tersebut`my-bucket-aps1`, log Anda

akan mencerminkan`GET` permintaan yang berhasil untuk`my-image.jpg` dari`my-bucket-aps1`, menggunakan nama host`mfzwi23gnjvgw.mrap.accesspoint.s3-global.amazonaws.com`.

> **Important**
>
> Titik Akses Multi-Wilayah tidak mengetahui isi data dari bucket yang mendasarinya. Oleh karena itu, bucket yang mendapat permintaan mungkin tidak berisi data yang diminta. Misalnya, jika Amazon S3 menentukan bahwa`my-bucket-euc1` bucket adalah yang terdekat, log Anda akan mencerminkan`GET` permintaan`my-image.jpg` dari yang gagal`my-bucket-euc1`, menggunakan nama host`mfzwi23gnjvgw.mrap.accesspoint.s3-global.amazonaws.com`. Jika permintaan dialihkan ke`my-bucket-usw2` sebaliknya, log Anda akan menunjukkan`GET` permintaan yang berhasil.

Untuk informasi lebih lanjut tentang pencatatan akses server Amazon S3, lihat Pencatatan permintaan dengan pencatatan akses server. Untuk informasi lebih lanjut tentang AWS CloudTrail, lihat Apa itu AWS CloudTrail? di Panduan Pengguna AWS CloudTrail.

Permintaan pemantauan dan pencatatan yang dibuat untuk operasi API manajemen Titik Akses Multi-Wilayah

Amazon S3 menyediakan beberapa operasi API untuk mengelola Titik Akses Multi-Wilayah, seperti`CreateMultiRegionAccessPoint` dan`GetMultiRegionAccessPointPolicy`. Saat Anda membuat permintaan ke operasi API ini dengan menggunakan AWS Command Line Interface (AWS CLI), AWS SDK, atau Amazon S3 REST API, Amazon S3 memproses permintaan ini secara asinkron. Asalkan Anda memiliki izin yang sesuai untuk permintaan tersebut, Amazon S3 mengembalikan token untuk permintaan ini. Anda dapat menggunakan token ini`DescribeAsyncOperation` untuk membantu Anda melihat status operasi asinkron yang sedang berlangsung. Amazon S3 memproses`DescribeAsyncOperation` permintaan secara sinkron. Untuk melihat status permintaan asinkron, Anda dapat menggunakan konsol Amazon S3, AWS CLI atau menggunakan konsol Amazon S3, atau atau API REST.

> **Note**
>
> Konsol hanya menampilkan status permintaan asinkron yang dibuat dalam 14 hari sebelumnya. Untuk melihat status permintaan yang lebih lama, gunakan, atau gunakan, atau

menggunakan, atau menggunakan, atau atau menggunakanAWS CLI, atau menggunakan, atau menggunakan, atau menggunakan, atau menggunakan, atau menggunakan, atau

Operasi manajemen asinkron dapat berada di salah satu dari beberapa kondisi:

NEW

Amazon S3 telah menerima permintaan dan sedang bersiap untuk melakukan operasi.

IN_PROGRESS

Amazon S3 saat ini sedang melakukan operasi.

SUCCESS

Operasi berhasil. Tanggapannya mencakup informasi yang relevan, seperti alias Titik Akses Multi-Wilayah untuk`CreateMultiRegionAccessPoint` permintaan.

FAILED

Operasi gagal. Respons menyertakan pesan kesalahan yang menjelaskan alasan kegagalan permintaan.

MenggunakanAWS CloudTrail dengan Titik Akses Multi-Wilayah

Anda dapat menggunakanAWS CloudTrail untuk melihat, mencari, mengunduh, mengarsipkan, menganalisis, dan merespons aktivitas akun di seluruhAWS infrastruktur Anda. Dengan Multi-Region Access Points dan CloudTrail logging, Anda dapat mengidentifikasi hal-hal berikut:

- Siapa atau apa yang mengambil tindakan mana
- Sumber daya mana yang ditindaklanjuti
- Saat acara terjadi
- Rincian lainnya mengenai acara tersebut

Anda dapat menggunakan informasi pencatatan ini untuk membantu Anda menganalisis dan merespons aktivitas yang terjadi melalui Titik Akses Multi-Wilayah Anda.

Cara mengaturAWS CloudTrail Titik Akses Multi-Wilayah

Untuk mengaktifkan CloudTrail logging untuk setiap operasi yang terkait dengan membuat atau memelihara Multi-Region Access Points, Anda harus mengkonfigurasi CloudTrail logging untuk merekam peristiwa di Wilayah AS Barat (Oregon). Anda harus menyiapkan konfigurasi pencatatan Anda dengan cara ini terlepas dari Wilayah mana Anda berada saat membuat permintaan, atau Wilayah mana yang didukung Titik Akses Multi-Wilayah. Semua permintaan untuk membuat atau memelihara titik akses multi-wilayah dialihkan melalui Wilayah US West (Oregon). Kami menyarankan Anda menambahkan Wilayah ini ke jejak yang ada atau membuat jejak baru yang berisi Wilayah ini dan semua Wilayah yang terkait dengan Titik Akses Multi-Wilayah.

Amazon S3 mencatat semua permintaan yang dibuat melalui Titik Akses Multi-Wilayah dan permintaan yang dibuat ke operasi API yang mengelola titik akses, sepertiCreateMultiRegionAccessPoint danGetMultiRegionAccessPointPolicy. Ketika Anda log permintaan ini melalui Multi-Region Access Point, mereka muncul diAWS CloudTrail log Anda dengan nama host dari Multi-Region Access Point. Misalnya, jika Anda membuat permintaan ke bucket melalui Multi-Region Access Point dengan aliasmfzwi23gnjvgw.mrap, entri dalam CloudTrail log akan memiliki nama hostmfzwi23gnjvgw.mrap.accesspoint.s3-global.amazonaws.com.

Titik Akses Multi-Wilayah mengarahkan permintaan ke bucket terdekatKarena perilaku ini, ketika Anda melihat CloudTrail log untuk Titik Akses Multi-Wilayah, Anda akan melihat permintaan dibuat ke bucket yang mendasarinya. Beberapa permintaan tersebut mungkin berupa permintaan langsung ke bucket yang tidak dialihkan melalui Multi-Region Access Point. Ingatlah fakta ini saat meninjau lalu lintas. Ketika bucket berada di Multi-Region Access Point, permintaan masih dapat dilakukan ke bucket itu secara langsung tanpa melalui Multi-Region Access Point.

Ada peristiwa asinkron yang terlibat dengan membuat dan mengelola Titik Akses Multi-Wilayah. Permintaan asinkron tidak memiliki peristiwa penyelesaian di CloudTrail log. Untuk informasi lebih lanjut tentang permintaan asinkron, lihatPermintaan pemantauan dan pencatatan yang dibuat untuk operasi API manajemen Titik Akses Multi-Wilayah.

Untuk informasi lebih lanjut tentang AWS CloudTrail, lihat Apa itu AWS CloudTrail? di Panduan Pengguna AWS CloudTrail.

Keamanan Amazon S3

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. Model tanggung jawab bersama menjelaskan hal ini sebagai keamanan cloud dan keamanan dalam cloud:

Keamanan cloud

AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Efektivitas keamanan kami diuji dan diverifikasi secara rutin oleh auditor pihak ketiga sebagai bagian dari program kepatuhan AWS. Untuk mempelajari tentang program kepatuhan yang berlaku untuk Amazon S3, lihat AWS Layanan dalam Cakupan Program Kepatuhan.

Keamanan cloud

Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor-faktor lain termasuk sensitivitas data Anda, persyaratan organisasi Anda, serta undang-undang dan peraturan yang berlaku. Untuk Amazon S3, tanggung jawab Anda mencakup area berikut:

- Mengelola data Anda, termasuk kepemilikan objek dan enkripsi.
- Mengklasifikasikan aset Anda.
- Mengelola akses ke data Anda menggunakan peran IAM dan konfigurasi layanan lainnya untuk menerapkan izin yang sesuai.
- Mengaktifkan kontrol detektif seperti atau Amazon GuardDuty untuk AWS CloudTrailAmazon S3.

Dokumentasi ini akan membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan Amazon S3. Topik berikut menunjukkan cara mengonfigurasi Amazon S3 untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga akan mempelajari cara menggunakan AWS layanan lain yang dapat membantu Anda memantau dan mengamankan sumber daya Amazon S3 Anda.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Topik

- Perlindungan data dalam Amazon S3
- Melindungi data dengan enkripsi
- Privasi lalu lintas antar jaringan
- AWS PrivateLink untuk Amazon S3
- Manajemen Akses
- Berbagi sumber daya lintas asal (CORS)
- Pencatatan log dan pemantauan di Amazon S3
- Validasi Kepatuhan untuk Amazon S3
- Ketahanan dalam Amazon S3
- Keamanan infrastruktur di Amazon S3
- Analisis konfigurasi dan kerentanan di Amazon S3
- Praktik Terbaik Keamanan untuk Amazon S3
- Memantau keamanan data dengan layanan AWS keamanan terkelola

Perlindungan data dalam Amazon S3

Amazon S3 menyediakan infrastruktur penyimpanan yang sangat tahan lama yang dirancang untuk penyimpanan data penting dan primer. S3 Standard, S3 Intelligent-Tiering, S3 Standard-IA, S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval, dan S3 Glacier Deep Archive menyimpan objek secara redundan di beberapa perangkat di minimal tiga Zona Ketersediaan dalam satu Wilayah AWS. Zona Ketersediaan adalah satu atau lebih pusat data diskret dengan daya redundan, jaringan, dan konektivitas dalam Wilayah AWS. Zona Ketersediaan secara fisik dipisahkan oleh jarak yang berarti, beberapa kilometer, dari Zona Ketersediaan lainnya, meskipun semuanya berada dalam jarak 100 km (60 mil) satu sama lain. Kelas penyimpanan S3 One Zone-IA menyimpan data secara redundan di beberapa perangkat dalam satu Zona Ketersediaan. Layanan ini dirancang

untuk menangani kegagalan perangkat secara bersamaan dengan mendeteksi dan memperbaiki redundansi yang hilang secara cepat, dan layanan ini juga secara rutin memverifikasi integritas data Anda menggunakan checksum.

Penyimpanan Amazon S3 standard menawarkan fitur berikut:

- Didukung dengan Perjanjian Tingkat Layanan Amazon S3.
- Dirancang untuk memberikan daya tahan 99,999999999% dan ketersediaan objek 99,99% pada tahun tertentu.
- S3 Standard, S3 Intelligent-Tiering, S3 Standard-IA, S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval, dan S3 Glacier Deep Archive semuanya dirancang untuk mempertahankan data jika terjadi peristiwa kehilangan Zona Ketersediaan Amazon S3 secara keseluruhan.

Amazon S3 selanjutnya melindungi data Anda menggunakan Penentuan Versi. Anda dapat menggunakan Penentuan Versi untuk melestarikan, mengambil, dan memulihkan setiap versi dari setiap objek yang disimpan di dalam bucket Amazon S3 Anda. Dengan Penentuan Versi, Anda dapat dengan mudah memulihkan dari tindakan pengguna yang tidak diinginkan, serta kegagalan aplikasi. Secara default, permintaan mengambil versi terbaru yang ditulis. Anda dapat mengambil versi lama objek dengan menentukan versi objek dalam permintaan.

Selain S3 Penentuan Versi, Anda juga dapat menggunakan Kunci Objek Amazon S3 dan Replikasi S3 untuk melindungi data Anda. Untuk informasi selengkapnya, lihat Tutorial: Melindungi data di Amazon S3 dari penghapusan yang tidak disengaja atau bug aplikasi menggunakan Penentuan Versi S3, Kunci Objek S3, dan Replikasi S3.

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensi dan mengatur akun pengguna individu dengan AWS Identity and Access Management, sehingga setiap pengguna hanya diberikan izin yang diperlukan untuk memenuhi tugas pekerjaan mereka.

Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-2 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Untuk informasi selengkapnya tentang titik akhir FIPS yang tersedia, lihat Federal Information Processing Standard (FIPS) 140-2.

Praktik terbaik keamanan berikut juga membahas perlindungan data di Amazon S3:

- Implement server-side encryption
- Enforce encryption of data in transit
- Consider using Macie with Amazon S3

- Identify and audit all your Amazon S3 buckets
- Monitor Amazon Web Services security advisories

Melindungi data dengan enkripsi

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Perlindungan data mengacu pada perlindungan data bergerak (saat melakukan perjalanan ke dan dari Amazon S3) dan saat diam (saat data disimpan di dalam disk di pusat data Amazon S3). Anda dapat melindungi data bergerak menggunakan Secure Socket Layer/Transport Layer Security (SSL/TLS) atau enkripsi di sisi klien. Untuk melindungi data diam di Amazon S3, Anda memiliki opsi berikut:

- Enkripsi sisi server — Amazon S3 mengenkripsi objek Anda sebelum menyimpannya di disk di pusat AWS data dan kemudian mendekripsi objek saat Anda mengunduhnya.

  Semua bucket Amazon S3 memiliki enkripsi yang dikonfigurasi secara default, dan semua objek baru yang diunggah ke bucket S3 secara otomatis dienkripsi saat sedang tidak aktif. Enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) adalah konfigurasi enkripsi default untuk setiap bucket di Amazon S3. Untuk menggunakan jenis enkripsi yang berbeda, Anda dapat menentukan jenis enkripsi di sisi server yang akan digunakan dalam permintaan PUT S3, atau Anda dapat mengatur konfigurasi enkripsi default di bucket tujuan.

  Jika Anda ingin menentukan jenis enkripsi yang berbeda dalam PUT permintaan Anda, Anda dapat menggunakan enkripsi sisi server dengan AWS Key Management Service () kunci (SSE-KMS AWS KMS), enkripsi sisi server dua lapis dengan kunci (DSSE-KMS), atau enkripsi sisi server dengan AWS KMS kunci yang disediakan pelanggan (SSE-C). Jika Anda ingin mengatur konfigurasi

enkripsi default yang berbeda di dalam bucket tujuan, Anda dapat menggunakan SSE-KMS atau DSSE-KMS.

Untuk informasi selengkapnya tentang setiap opsi untuk enkripsi sisi server, lihat. Melindungi data dengan enkripsi di sisi klien

Untuk mengonfigurasi enkripsi di sisi server, lihat:

- Menentukan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3)
- Menentukan enkripsi di sisi server dengan AWS KMS (SSE-KMS)
- the section called “Menentukan SSE-KMS”
- Menentukan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C)

- Enkripsi di sisi klien–Anda mengenkripsi data sisi klien dan mengunggah data yang dienkripsi ke Amazon S3. Dalam hal ini, Anda mengelola proses enkripsi, kunci enkripsi, dan alat terkait.

  Untuk mengonfigurasi enkripsi di sisi klien, lihat Melindungi data menggunakan enkripsi di sisi klien.

Untuk melihat persentase byte penyimpanan yang dienkripsi, Anda dapat menggunakan metrik Lensa Penyimpanan Amazon S3. Lensa Penyimpanan S3 adalah fitur analisis penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan objek. Untuk informasi selengkapnya, lihat Menilai aktivitas penyimpanan dan penggunaan Anda dengan Lensa Penyimpanan S3. Untuk daftar lengkap metrik, lihat Glosarium metrik Lensa Penyimpanan S3.

Untuk informasi selengkapnya tentang enkripsi di sisi server dan enkripsi di sisi klien, tinjau topik berikut.

Topik

- Melindungi data dengan enkripsi di sisi klien
- Melindungi data menggunakan enkripsi di sisi klien

Melindungi data dengan enkripsi di sisi klien

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari

2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Enkripsi di sisi server adalah enkripsi data di tempat tujuan oleh aplikasi atau layanan yang menerimanya. Amazon S3 mengenkripsi data Anda pada tingkat objek saat menulisnya ke disk di pusat AWS data dan mendekripsi untuk Anda saat Anda mengaksesnya. Selama Anda mengautentikasi permintaan dan telah memiliki izin akses, tidak ada perbedaan dalam cara mengakses objek terenkripsi atau tidak terenkripsi. Misalnya, jika Anda berbagi objek menggunakan URL yang ditandatangani sebelumnya, URL tersebut bekerja dengan cara yang sama untuk objek terenkripsi maupun yang tidak terenkripsi. Selain itu, saat Anda mencantumkan objek di bucket, operasi daftar API menampilkan daftar semua objek, terlepas dari apakah objek tersebut dienkripsi atau tidak.

Semua bucket Amazon S3 memiliki enkripsi yang dikonfigurasi secara default, dan semua objek baru yang diunggah ke bucket S3 secara otomatis dienkripsi saat sedang tidak aktif. Enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) adalah konfigurasi enkripsi default untuk setiap bucket di Amazon S3. Untuk menggunakan jenis enkripsi yang berbeda, Anda dapat menentukan jenis enkripsi di sisi server yang akan digunakan dalam permintaan PUT S3, atau Anda dapat mengatur konfigurasi enkripsi default di bucket tujuan.

Jika Anda ingin menentukan jenis enkripsi yang berbeda dalam PUT permintaan Anda, Anda dapat menggunakan enkripsi sisi server dengan AWS Key Management Service () kunci (SSE-KMS AWS KMS), enkripsi sisi server dua lapis dengan kunci (DSSE-KMS), atau enkripsi sisi server dengan AWS KMS kunci yang disediakan pelanggan (SSE-C). Jika Anda ingin mengatur konfigurasi enkripsi default yang berbeda di dalam bucket tujuan, Anda dapat menggunakan SSE-KMS atau DSSE-KMS.

> **Note**
>
> Anda tidak dapat menerapkan berbagai jenis enkripsi di sisi server ke objek yang sama secara bersamaan.

Jika Anda perlu mengenkripsi objek yang ada, gunakan Operasi Batch S3 dan Inventaris S3. Untuk informasi selengkapnya, lihat Mengenkripsi objek dengan Operasi Batch Amazon S3 dan Melakukan operasi batch berskala besar pada objek Amazon S3.

Anda memiliki empat opsi yang saling eksklusif untuk enkripsi di sisi server, tergantung pada cara Anda mengelola kunci enkripsi dan jumlah lapisan enkripsi yang ingin Anda terapkan.

Enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3)

Semua bucket Amazon S3 memiliki enkripsi yang dikonfigurasi secara default. Opsi default untuk enkripsi di sisi server adalah dengan kunci terkelola Amazon S3 (SSE-S3). Setiap objek dienkripsi dengan kunci unik. Sebagai pelindung tambahan, SSE-S3 mengenkripsi kunci itu sendiri dengan kunci root yang secara rutin diputar. SSE-S3 menggunakan salah satu penyandian blok terkuat yang ada, Advanced Encryption Standard 256-bit (AES-256), untuk mengenkripsi data Anda. Untuk informasi selengkapnya, lihat Menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3).

Enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS)

Enkripsi sisi server dengan AWS KMS keys (SSE-KMS) disediakan melalui integrasi layanan dengan Amazon S3. AWS KMS Dengan AWS KMS, Anda memiliki kontrol lebih besar atas kunci Anda. Misalnya, Anda dapat melihat kunci terpisah, mengedit kebijakan kontrol, dan mengikuti kunci di AWS CloudTrail. Selain itu, Anda dapat membuat dan mengelola CMK atau menggunakan Kunci yang dikelola AWS yang unik bagi Anda, layanan Anda, dan Wilayah Anda. Untuk informasi selengkapnya, lihat Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS).

Enkripsi sisi server dua lapis dengan kunci () (DSSE-KMS AWS Key Management Service)AWS KMS

Enkripsi sisi server dual-layer dengan AWS KMS keys (DSSE-KMS) mirip dengan SSE-KMS, tetapi DSSE-KMS menerapkan dua lapisan individual enkripsi tingkat objek, bukan satu lapisan. Karena kedua lapisan enkripsi diterapkan ke objek di sisi server, Anda dapat menggunakan berbagai alat Layanan AWS dan untuk menganalisis data di S3 sambil menggunakan metode enkripsi yang dapat memenuhi persyaratan kepatuhan Anda. Untuk informasi selengkapnya, lihat Menggunakan enkripsi sisi server dual-layer dengan kunci (DSSE-KMS) AWS KMS.

Enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C)

Dengan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C), Anda mengelola kunci enkripsi, dan Amazon S3 mengelola enkripsi saat menulis ke disk dan dekripsi saat Anda

mengakses objek Anda. Untuk informasi selengkapnya, lihat Menggunakan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C).

Amazon S3 sekarang secara otomatis mengenkripsi semua objek baru

Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. SSE-S3, yang menggunakan Standar Enkripsi Lanjutan 256-bit (AES-256), secara otomatis diterapkan ke semua bucket baru dan ke bucket S3 yang ada yang belum memiliki enkripsi default yang dikonfigurasi. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di () dan SDK. AWS Command Line Interface AWS CLI AWS

Bagian berikut menjawab pertanyaan tentang pembaruan ini.

Apakah Amazon S3 mengubah pengaturan enkripsi default untuk bucket saya yang sudah ada yang sudah memiliki enkripsi default yang dikonfigurasi?

Tidak. Tidak ada perubahan pada konfigurasi enkripsi default untuk bucket yang sudah ada yang sudah memiliki enkripsi SSE-S3 atau sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS) yang dikonfigurasi. Untuk informasi lebih lanjut tentang cara mengatur perilaku enkripsi default untuk bucket, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3. Untuk informasi selengkapnya tentang pengaturan enkripsi SSE-S3 dan SSE-KMS, lihat Melindungi data dengan enkripsi di sisi klien.

Apakah enkripsi default diaktifkan pada bucket saya yang sudah ada yang enkripsi defaultnya tidak dikonfigurasi?

Ya. Amazon S3 sekarang mengonfigurasi enkripsi default pada semua bucket tidak terenkripsi yang sudah ada untuk menerapkan enkripsi di sisi server dengan kunci terkelola S3 (SSE-S3) sebagai tingkat dasar enkripsi untuk objek baru yang diunggah ke bucket ini. Objek yang sudah ada dalam bucket tidak terenkripsi tidak akan dienkripsi secara otomatis.

Bagaimana saya bisa melihat status enkripsi default dari unggahan objek baru?

Saat ini, Anda dapat melihat status enkripsi default dari unggahan objek baru di AWS CloudTrail log, Inventaris S3, dan Lensa Penyimpanan S3, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di () dan SDK. AWS Command Line Interface AWS CLI AWS

- Untuk melihat CloudTrail acara Anda, lihat Melihat CloudTrail peristiwa di CloudTrail konsol di Panduan AWS CloudTrail Pengguna. CloudTrail log menyediakan pelacakan API untuk `PUT` dan `POST` permintaan ke Amazon S3. Saat enkripsi default digunakan untuk mengenkripsi objek di bucket Anda, CloudTrail log untuk `PUT` dan permintaan `POST` API akan menyertakan bidang berikut sebagai pasangan nama-nilai:. `"SSEApplied":"Default_SSE_S3"`
- Untuk melihat status enkripsi otomatis dari unggahan objek baru di Inventaris S3, konfigurasikan laporan Inventaris S3 untuk menyertakan bidang metadata Enkripsi, lalu lihat status enkripsi untuk setiap objek baru dalam laporan. Untuk informasi selengkapnya, lihat Mengatur Inventaris Amazon S3.
- Untuk melihat status enkripsi otomatis untuk unggahan objek baru di Lensa Penyimpanan S3, konfigurasikan dasbor Lensa Penyimpanan S3 dan lihat metrik Byte terenkripsi dan Jumlah objek terenkripsi dalam kategori Perlindungan data pada dasbor. Untuk informasi selengkapnya, lihat Membuat dasbor Lensa Penyimpanan Amazon S3 dan Melihat metrik S3 Storage Lens di dasbor.
- Untuk melihat status enkripsi tingkat bucket otomatis di konsol Amazon S3, periksa Enkripsi default bucket Amazon S3 Anda di konsol Amazon S3. Untuk informasi selengkapnya, lihat Mengonfigurasi enkripsi default.
- Untuk melihat status enkripsi otomatis sebagai header respons Amazon S3 API tambahan di AWS Command Line Interface (AWS CLI) dan AWS SDK, periksa header respons `x-amz-server-side-encryption` saat Anda menggunakan API tindakan objek, seperti dan. PutObjectGetObject

Apa yang harus saya lakukan untuk memanfaatkan perubahan ini?

Anda tidak perlu melakukan perubahan apa pun pada aplikasi Anda yang ada. Karena enkripsi default diaktifkan untuk semua bucket Anda, semua objek baru yang diunggah ke Amazon S3 secara otomatis dienkripsi.

Bisakah saya menonaktifkan enkripsi untuk objek baru yang ditulis ke bucket saya?

Tidak. SSE-S3 adalah enkripsi tingkat dasar baru yang diterapkan pada semua objek baru yang diunggah ke bucket Anda. Anda tidak dapat lagi menonaktifkan enkripsi untuk unggahan objek baru.

Apakah biaya saya akan terpengaruh?

Tidak. Enkripsi default dengan SSE-S3 tersedia tanpa biaya tambahan. Anda akan ditagih untuk penyimpanan, permintaan, dan fitur S3 lainnya, seperti biasa. Untuk informasi harga, lihat Harga Amazon S3.

Akankah Amazon S3 mengenkripsi objek saya yang tidak terenkripsi?

Tidak. Mulai 5 Januari 2023, Amazon S3 hanya secara otomatis mengenkripsi unggahan objek baru. Untuk mengenkripsi objek yang ada, Anda dapat menggunakan operasi batch S3 untuk membuat salinan terenkripsi objek Anda. Salinan terenkripsi ini akan mempertahankan data dan nama objek yang ada dan akan dienkripsi dengan menggunakan kunci enkripsi yang Anda tentukan. Untuk detail selengkapnya, lihat Mengenkripsi objek dengan Operasi Batch Amazon S3 dalam AWS Blog Penyimpanan.

Saya tidak mengaktifkan enkripsi untuk bucket saya sebelum rilis ini. Apakah saya perlu mengubah cara saya mengakses objek?

Tidak. Enkripsi default dengan SSE-S3 secara otomatis mengenkripsi data Anda seperti yang tertulis ke Amazon S3 dan mendekripsinya untuk Anda saat Anda mengaksesnya. Tidak ada perubahan dalam cara Anda mengakses objek yang dienkripsi secara otomatis.

Apakah saya perlu mengubah cara saya mengakses objek terenkripsi di sisi klien?

Tidak. Semua objek terenkripsi di sisi klien yang dienkripsi sebelum diunggah ke Amazon S3 tiba sebagai objek teks sandi terenkripsi dalam Amazon S3. Objek ini sekarang akan memiliki lapisan enkripsi SSE-S3. Beban kerja Anda yang menggunakan objek yang terenkripsi di sisi klien tidak akan memerlukan perubahan apa pun pada layanan klien atau pengaturan otorisasi Anda.

> **Note**
>
> HashiCorp Pengguna Terraform yang tidak menggunakan versi AWS Penyedia yang diperbarui mungkin melihat penyimpangan yang tidak terduga setelah membuat bucket S3 baru tanpa konfigurasi enkripsi yang ditentukan pelanggan. Untuk menghindari penyimpangan ini, perbarui versi AWS Penyedia Terraform Anda ke salah satu versi berikut: 4.x rilis apa pun,, 3.76.1 atau. 2.70.4

Menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3)

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS

CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Semua unggahan objek baru ke bucket Amazon S3 dienkripsi secara default dengan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3).

Enkripsi sisi-server melindungi data diam. Amazon S3 mengenkripsi setiap objek dengan kunci unik. Sebagai perlindungan tambahan, ia mengenkripsi kunci itu sendiri dengan kunci yang diputar secara berkala. enkripsi di sisi server Amazon S3 menggunakan Advanced Encryption Standard Galois/Counter Mode (AES-GCM) 256 bit untuk mengenkripsi semua objek yang diunggah.

Tidak ada biaya tambahan untuk menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3). Namun, permintaan untuk mengonfigurasi fitur enkripsi default dikenakan biaya permintaan Amazon S3 standar. Untuk informasi tentang harga, lihat Harga Amazon S3.

Jika Anda mengharuskan unggahan data dienkripsi hanya menggunakan kunci terkelola Amazon S3, Anda dapat menggunakan kebijakan bucket berikut. Misalnya, kebijakan bucket berikut menolak izin untuk mengunggah objek kecuali jika permintaan mencakup header `x-amz-server-side-encryption` untuk meminta enkripsi di sisi server:

```
{
  "Version": "2012-10-17",
  "Id": "PutObjectPolicy",
  "Statement": [
    {
      "Sid": "DenyObjectsThatAreNotSSES3",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::example-s3-bucket/*",
      "Condition": {
        "StringNotEquals": {
          "s3:x-amz-server-side-encryption": "AES256"
        }
      }
    }
   ]
}
```

> **Note**
>
> Enkripsi di sisi server hanya mengenkripsi data objek, bukan metadata objek.

Dukungan API untuk enkripsi di sisi server

Semua bucket Amazon S3 memiliki enkripsi yang dikonfigurasi secara default, dan semua objek baru yang diunggah ke bucket S3 secara otomatis dienkripsi saat sedang tidak aktif. Enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) adalah konfigurasi enkripsi default untuk setiap bucket di Amazon S3. Untuk menggunakan jenis enkripsi yang berbeda, Anda dapat menentukan jenis enkripsi di sisi server yang akan digunakan dalam permintaan PUT S3, atau Anda dapat mengatur konfigurasi enkripsi default di bucket tujuan.

Jika Anda ingin menentukan jenis enkripsi yang berbeda dalam PUT permintaan Anda, Anda dapat menggunakan enkripsi sisi server dengan AWS Key Management Service () kunci (SSE-KMS AWS KMS), enkripsi sisi server dua lapis dengan kunci (DSSE-KMS), atau enkripsi sisi server dengan AWS KMS kunci yang disediakan pelanggan (SSE-C). Jika Anda ingin mengatur konfigurasi enkripsi default yang berbeda di dalam bucket tujuan, Anda dapat menggunakan SSE-KMS atau DSSE-KMS.

Untuk mengonfigurasi enkripsi di sisi server dengan menggunakan API REST pembuatan objek, Anda harus menyediakan header permintaan `x-amz-server-side-encryption`. Untuk informasi tentang API REST, lihat Penggunaan API REST.

API Amazon S3 berikut mendukung header ini:

- Operasi PUT–Tentukan header permintaan saat mengunggah data menggunakan API PUT. Untuk informasi lebih lanjut, lihat PUT Objek.
- Mulai Unggahan Multibagian–Menentukan header dalam permintaan awal saat mengunggah objek besar menggunakan operasi API multibagian. Untuk informasi lebih lanjut, lihat Mulai Unggahan Multibagian.
- Operasi COPY–Saat Anda menyalin objek, Anda memiliki objek sumber dan objek target. Untuk informasi lebih lanjut, lihat PUT Objek-Salin.

> **Note**
>
> Saat menggunakan operasi POST untuk mengunggah objek, alih-alih memberikan header permintaan, Anda memberikan informasi yang sama di kolom formulir. Untuk informasi lebih lanjut, lihat Objek POST.

AWS SDK juga menyediakan API pembungkus yang dapat Anda gunakan untuk meminta enkripsi sisi server. Anda juga dapat menggunakan AWS Management Console untuk mengunggah objek dan meminta enkripsi sisi server.

Untuk informasi lebih umum, lihat AWS KMS konsep dalam AWS Key Management Service Panduan Pengembang.

Topik

- Menentukan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3)

Menentukan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3)

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Semua bucket Amazon S3 memiliki enkripsi yang dikonfigurasi secara default, dan semua objek baru yang diunggah ke bucket S3 secara otomatis dienkripsi saat sedang tidak aktif. Enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) adalah konfigurasi enkripsi default untuk setiap bucket di Amazon S3. Untuk menggunakan jenis enkripsi yang berbeda, Anda dapat menentukan jenis enkripsi di sisi server yang akan digunakan dalam permintaan PUT S3, atau Anda dapat mengatur konfigurasi enkripsi default di bucket tujuan.

Jika Anda ingin menentukan jenis enkripsi yang berbeda dalam PUT permintaan Anda, Anda dapat menggunakan enkripsi sisi server dengan AWS Key Management Service () kunci (SSE-KMS AWS KMS), enkripsi sisi server dua lapis dengan kunci (DSSE-KMS), atau enkripsi sisi server dengan AWS KMS kunci yang disediakan pelanggan (SSE-C). Jika Anda ingin mengatur konfigurasi enkripsi default yang berbeda di dalam bucket tujuan, Anda dapat menggunakan SSE-KMS atau DSSE-KMS.

Anda dapat menentukan SSE-S3 dengan menggunakan konsol S3, REST API, AWS SDK, dan (). AWS Command Line Interface AWS CLI Untuk informasi selengkapnya, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3.

Menggunakan konsol S3

Topik ini menjelaskan cara mengatur atau mengubah jenis enkripsi objek dengan menggunakan AWS Management Console. Saat menyalin objek dengan menggunakan konsol, Amazon S3 menyalin objek seperti apa adanya. Itu berarti bahwa jika objek sumber dienkripsi, objek target juga dienkripsi. Anda dapat menggunakan konsol untuk menambahkan atau mengubah enkripsi untuk suatu objek.

> **Note**
>
> - Jika Anda mengubah enkripsi objek, sebuah objek baru akan dibuat untuk menggantikan objek yang lama. Jika Penentuan Versi S3 diaktifkan, versi baru objek akan dibuat, dan objek yang sudah ada menjadi versi yang lebih lama. Peran yang mengubah properti juga menjadi pemilik objek baru (atau versi objek).
> - Jika Anda mengubah jenis enkripsi untuk objek yang memiliki tag yang ditentukan pengguna, Anda harus memiliki izin. `s3:GetObjectTagging` Jika Anda mengubah jenis enkripsi untuk objek yang tidak memiliki tag yang ditentukan pengguna tetapi berukuran lebih dari 16 MB, Anda juga harus memiliki izin. `s3:GetObjectTagging`
>
> Jika kebijakan bucket tujuan menolak `s3:GetObjectTagging` tindakan, jenis enkripsi untuk objek akan diperbarui, tetapi tag yang ditentukan pengguna akan dihapus dari objek, dan Anda akan menerima kesalahan.

Untuk mengubah enkripsi untuk objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di panel navigasi kiri, pilih Bucket.
3. Di dalam daftar Bucket, pilih nama bucket yang berisi objek.
4. Di dalam daftar Objek, pilih nama objek yang ingin Anda tambahkan atau ubah enkripsinya.

   Halaman detail objek muncul, dengan beberapa bagian yang menampilkan properti untuk objek Anda.
5. Pilih tab Properti.
6. Gulir ke bawah ke bagian Pengaturan enkripsi di sisi server, lalu pilih Edit.
7. Di bawah Pengaturan enkripsi, pilih Gunakan pengaturan enkripsi default bucket, atau Timpa pengaturan enkripsi default bucket.
8. Jika Anda memilih Timpa pengaturan bucket untuk enkripsi default, konfigurasikan pengaturan enkripsi berikut.

   - Di bawah Jenis enkripsi, pilih Kunci yang dikelola Amazon S3 (SSE-S3). SSE-S3 menggunakan salah satu penyandian blok terkuat—Advanced Encryption Standard 256-bit (AES-256) untuk mengenkripsi setiap objek. Untuk informasi selengkapnya, lihat Menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3).
9. Pilih Simpan perubahan.

> **Note**
>
> Tindakan ini menerapkan enkripsi untuk semua objek yang ditentukan. Saat mengenkripsi folder, tunggu hingga operasi penyimpanannya selesai sebelum menambahkan objek baru ke folder tersebut.

Penggunaan API REST

Saat pembuatan objek—yaitu, saat Anda mengunggah objek baru atau membuat salinan objek yang sudah ada—Anda dapat menentukan apakah Anda ingin Amazon S3 mengenkripsi data Anda dengan kunci yang dikelola Amazon S3 (SSE-S3) dengan menambahkan header ke permintaan `x-amz-server-side-encryption`. Mengatur nilai header ke algoritma enkripsi `AES256`, yang didukung Amazon S3. Amazon S3 mengonfirmasi bahwa objek Anda disimpan dengan SSE-S3 dengan mengembalikan header respons `x-amz-server-side-encryption`.

Operasi API unggahan REST berikut ini menerima header permintaan `x-amz-server-side-encryption`.

- PUT Objek
- PUT Objek-Salin
- Objek POST
- Mulai Unggahan Multibagian

Saat mengunggah objek besar menggunakan operasi API unggahan multibagian, Anda dapat menentukan enkripsi di sisi server dengan menambahkan `x-amz-server-side-encryption` header ke permintaan Mulai Pengunggahan Multibagian. Saat Anda menyalin objek yang ada, terlepas dari apakah objek sumbernya dienkripsi atau tidak, objek tujuan tidak dienkripsi kecuali jika Anda secara eksplisit meminta enkripsi di sisi server.

Header respons dari operasi API REST berikut mengembalikan `x-amz-server-side-encryption` header saat objek disimpan menggunakan SSE-S3.

- PUT Objek
- PUT Objek-Salin
- Objek POST
- Mulai Unggahan Multibagian
- Unggah Bagian
- Unggah Bagian-Salin
- Selesaikan Unggahan Multibagian
- Dapatkan Objek
- Head Objek

> **Note**
>
> Jangan mengirim header permintaan enkripsi untuk `GET` permintaan dan `HEAD` minta jika objek Anda menggunakan SSE-S3, atau Anda akan mendapatkan kesalahan kode status HTTP 400 (Permintaan Buruk).

Menggunakan AWS SDK

Saat menggunakan AWS SDK, Anda dapat meminta Amazon S3 untuk menggunakan enkripsi sisi server dengan kunci enkripsi terkelola Amazon S3 (SSE-S3). Bagian ini memberikan contoh penggunaan AWS SDK dalam berbagai bahasa. Untuk informasi tentang SDK lainnya, buka Kode Sampel dan Pustaka.

Java

Ketika Anda menggunakan AWS SDK for Java untuk meng-upload objek, Anda dapat menggunakan SSE-S3 untuk mengenkripsi itu. Untuk meminta enkripsi di sisi server, gunakan properti `ObjectMetadata` dari `PutObjectRequest` untuk menetapkan header permintaan `x-amz-server-side-encryption`. Saat Anda memanggil `putObject()` metode dari `AmazonS3Client`, Amazon S3 mengenkripsi dan menyimpan data.

Anda juga dapat meminta enkripsi SSE-S3 saat mengunggah objek menggunakan operasi API pengunggahan multibagian:

- Saat menggunakan operasi API pengunggahan multibagian tingkat tinggi, Anda menggunakan metode `TransferManager` untuk menerapkan enkripsi di sisi server ke objek saat Anda mengunggahnya. Anda dapat menggunakan metode pengunggahan apa saja yang menggunakan `ObjectMetadata` sebagai parameter. Untuk informasi selengkapnya, lihat Pengunggahan objek menggunakan unggahan multibagian.
- Saat menggunakan operasi API pengunggahan multibagian tingkat rendah, Anda menentukan enkripsi di sisi server saat Anda memulai pengunggahan multibagian. Anda menambahkan properti `ObjectMetadata` dengan memanggil metode `InitiateMultipartUploadRequest.setObjectMetadata()`. Untuk informasi selengkapnya, lihat Menggunakan AWS SDK (API tingkat rendah).

Anda tidak dapat langsung mengubah status enkripsi suatu objek (mengenkripsi objek yang tidak dienkripsi atau mendekripsi objek yang dienkripsi). Untuk mengubah status enkripsi objek, Anda membuat salinan objek, dengan menentukan status enkripsi yang diinginkan untuk salinan, lalu hapus objek asli. Amazon S3 mengenkripsi objek yang disalin hanya jika Anda secara eksplisit meminta enkripsi di sisi server. Untuk meminta enkripsi objek yang disalin melalui API Java, gunakan properti `ObjectMetadata` untuk menentukan enkripsi di sisi server dalam `CopyObjectRequest`.

Example Contoh

Contoh berikut menunjukkan cara menetapkan enkripsi di sisi server menggunakan AWS SDK for Java. Itu menunjukkan cara melakukan tugas berikut:

- Unggah objek baru menggunakan SSE-S3.
- Ubah status enkripsi objek (dalam contoh ini, mengenkripsi objek yang tidak dienkripsi sebelumnya) dengan membuat salinan objek.
- Periksa status enkripsi objek.

Untuk informasi lebih lanjut tentang enkripsi di sisi server, lihat Penggunaan API REST. Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.internal.SSEResultBase;
import com.amazonaws.services.s3.model.*;

import java.io.ByteArrayInputStream;

public class SpecifyServerSideEncryption {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyNameToEncrypt = "*** Key name for an object to upload and encrypt
 ***";
        String keyNameToCopyAndEncrypt = "*** Key name for an unencrypted object to
 be encrypted by copying ***";
        String copiedObjectKeyName = "*** Key name for the encrypted copy of the
 unencrypted object ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
```

```
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Upload an object and encrypt it with SSE.
            uploadObjectWithSSEEncryption(s3Client, bucketName, keyNameToEncrypt);

            // Upload a new unencrypted object, then change its encryption state
            // to encrypted by making a copy.
            changeSSEEncryptionStatusByCopying(s3Client,
                    bucketName,
                    keyNameToCopyAndEncrypt,
                    copiedObjectKeyName);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    private static void uploadObjectWithSSEEncryption(AmazonS3 s3Client, String
bucketName, String keyName) {
        String objectContent = "Test object encrypted with SSE";
        byte[] objectBytes = objectContent.getBytes();

        // Specify server-side encryption.
        ObjectMetadata objectMetadata = new ObjectMetadata();
        objectMetadata.setContentLength(objectBytes.length);

objectMetadata.setSSEAlgorithm(ObjectMetadata.AES_256_SERVER_SIDE_ENCRYPTION);
        PutObjectRequest putRequest = new PutObjectRequest(bucketName,
                keyName,
                new ByteArrayInputStream(objectBytes),
                objectMetadata);

        // Upload the object and check its encryption status.
        PutObjectResult putResult = s3Client.putObject(putRequest);
        System.out.println("Object \"" + keyName + "\" uploaded with SSE.");
        printEncryptionStatus(putResult);
    }
```

```
    private static void changeSSEEncryptionStatusByCopying(AmazonS3 s3Client,
            String bucketName,
            String sourceKey,
            String destKey) {
        // Upload a new, unencrypted object.
        PutObjectResult putResult = s3Client.putObject(bucketName, sourceKey,
 "Object example to encrypt by copying");
        System.out.println("Unencrypted object \"" + sourceKey + "\" uploaded.");
        printEncryptionStatus(putResult);

        // Make a copy of the object and use server-side encryption when storing the
        // copy.
        CopyObjectRequest request = new CopyObjectRequest(bucketName,
                sourceKey,
                bucketName,
                destKey);
        ObjectMetadata objectMetadata = new ObjectMetadata();

 objectMetadata.setSSEAlgorithm(ObjectMetadata.AES_256_SERVER_SIDE_ENCRYPTION);
        request.setNewObjectMetadata(objectMetadata);

        // Perform the copy operation and display the copy's encryption status.
        CopyObjectResult response = s3Client.copyObject(request);
        System.out.println("Object \"" + destKey + "\" uploaded with SSE.");
        printEncryptionStatus(response);

        // Delete the original, unencrypted object, leaving only the encrypted copy
 in
        // Amazon S3.
        s3Client.deleteObject(bucketName, sourceKey);
        System.out.println("Unencrypted object \"" + sourceKey + "\" deleted.");
    }

    private static void printEncryptionStatus(SSEResultBase response) {
        String encryptionStatus = response.getSSEAlgorithm();
        if (encryptionStatus == null) {
            encryptionStatus = "Not encrypted with SSE";
        }
        System.out.println("Object encryption status is: " + encryptionStatus);
    }
}
```

.NET

Saat mengunggah sebuah objek, Anda dapat mengarahkan Amazon S3 untuk mengenkripsinya. Untuk mengubah status enkripsi objek yang sudah ada, Anda membuat salinan objek dan menghapus objek sumber. Secara default, operasi penyalinan mengenkripsi target hanya jika Anda secara eksplisit meminta enkripsi di sisi server objek target. Untuk menentukan SSE-S3 di `CopyObjectRequest`, tambahkan yang berikut ini:

```
ServerSideEncryptionMethod = ServerSideEncryptionMethod.AES256
```

Untuk sampel kerja tentang cara menyalin objek, lihat Menggunakan AWS SDK.

Contoh berikut mengunggah objek. Dalam permintaan, contoh tersebut mengarahkan Amazon S3 untuk mengenkripsi objek. Contoh tersebut kemudian mengambil metadata objek dan memverifikasi metode enkripsi yang digunakan. Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class SpecifyServerSideEncryptionTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** key name for object created ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            WritingAnObjectAsync().Wait();
        }

        static async Task WritingAnObjectAsync()
```

```
        {
            try
            {
                var putRequest = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    ContentBody = "sample text",
                    ServerSideEncryptionMethod = ServerSideEncryptionMethod.AES256
                };

                var putResponse = await client.PutObjectAsync(putRequest);

                // Determine the encryption state of an object.
                GetObjectMetadataRequest metadataRequest = new
 GetObjectMetadataRequest
                {
                    BucketName = bucketName,
                    Key = keyName
                };
                GetObjectMetadataResponse response = await
 client.GetObjectMetadataAsync(metadataRequest);
                ServerSideEncryptionMethod objectEncryption =
 response.ServerSideEncryptionMethod;

                Console.WriteLine("Encryption method used: {0}",
 objectEncryption.ToString());
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered ***. Message:'{0}' when writing
 an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

PHP

Topik ini menunjukkan cara menggunakan kelas dari versi 3 AWS SDK for PHP untuk menambahkan SSE-S3 ke objek yang Anda unggah ke Amazon S3. Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

Untuk mengunggah objek ke Amazon S3, gunakan metode Aws\S3\S3Client::putObject(). Untuk menambahkan `x-amz-server-side-encryption` header permintaan ke permintaan pengunggahan, tentukan parameter `ServerSideEncryption` dengan nilai `AES256`, seperti yang ditunjukkan dalam contoh kode berikut. Untuk informasi lebih lanjut tentang permintaan enkripsi di sisi server, lihat Penggunaan API REST.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

// $filepath should be an absolute path to a file on disk.
$filepath = '*** Your File Path ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Upload a file with server-side encryption.
$result = $s3->putObject([
    'Bucket'               => $bucket,
    'Key'                  => $keyname,
    'SourceFile'           => $filepath,
    'ServerSideEncryption' => 'AES256',
]);
```

Sebagai respons, Amazon S3 mengembalikan header `x-amz-server-side-encryption` dengan nilai algoritma enkripsi yang digunakan untuk mengenkripsi data objek Anda.

Saat mengunggah objek besar dengan menggunakan operasi API unggahan multibagian, Anda dapat menentukan SSE-S3 untuk objek yang Anda unggah, sebagai berikut ini:

- Saat Anda menggunakan operasi API unggahan multibagian tingkat rendah, tentukan enkripsi sisi server saat Anda memanggil metode Aws\ S3\ S3Client:: (). createMultipartUpload Untuk menambahkan header permintaan `x-amz-server-side-encryption` untuk permintaan Anda, tentukan parameter `array` kunci `ServerSideEncryption` dengan nilai `AES256`. Untuk informasi selengkapnya tentang operasi API unggahan multibagian tingkat rendah, lihat Menggunakan AWS SDK (API tingkat rendah).
- Saat Anda menggunakan operasi API unggahan multibagian tingkat tinggi, tentukan enkripsi sisi server dengan menggunakan `ServerSideEncryption` parameter operasi API. CreateMultipartUpload Untuk contoh penggunaan metode `setOption()` dengan operasi API unggahan multibagian tingkat tinggi, lihat Pengunggahan objek menggunakan unggahan multibagian.

Untuk menentukan status enkripsi objek yang ada, ambil metadata objek dengan memanggil metode Aws\S3\S3Client::headObject() seperti yang ditunjukkan dalam contoh kode PHP berikut.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';
$keyname = '*** Your Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Check which server-side encryption algorithm is used.
$result = $s3->headObject([
    'Bucket' => $bucket,
    'Key'    => $keyname,
]);
echo $result['ServerSideEncryption'];
```

Untuk mengubah status enkripsi objek yang sudah ada, buat salinan objek menggunakan metode Aws\S3\S3Client::copyObject() dan menghapus objek sumber. Secara default, `copyObject()` tidak mengenkripsi target kecuali jika Anda secara eksplisit meminta enkripsi di sisi server objek tujuan menggunakan parameter `ServerSideEncryption` dengan nilai `AES256`. Contoh kode

PHP berikut membuat salinan objek dan menambahkan enkripsi server-side ke objek yang tersalin.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;

$sourceBucket = '*** Your Source Bucket Name ***';
$sourceKeyname = '*** Your Source Object Key ***';

$targetBucket = '*** Your Target Bucket Name ***';
$targetKeyname = '*** Your Target Object Key ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);

// Copy an object and add server-side encryption.
$s3->copyObject([
    'Bucket'               => $targetBucket,
    'Key'                  => $targetKeyname,
    'CopySource'           => "$sourceBucket/$sourceKeyname",
    'ServerSideEncryption' => 'AES256',
]);
```

Untuk informasi selengkapnya, lihat topik berikut.

- AWS SDK for PHP untuk Amazon S3 Aws\ S3\ S3Kelas Klien
- Dokumentasi AWS SDK for PHP

Ruby

Saat menggunakan AWS SDK for Ruby untuk mengunggah objek, Anda dapat menentukan bahwa objek disimpan terenkripsi saat istirahat dengan SSE-S3. Saat Anda membaca kembali, objek tersebut akan didekripsi secara otomatis.

Contoh AWS SDK for Ruby Versi 3 berikut menunjukkan cara menentukan bahwa file yang diunggah ke Amazon S3 dienkripsi saat istirahat.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectPutSseWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  def put_object_encrypted(object_content, encryption)
    @object.put(body: object_content, server_side_encryption: encryption)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't put your content to #{object.key}. Here's why: #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-encrypted-content"
  object_content = "This is my super-secret content."
  encryption = "AES256"

  wrapper = ObjectPutSseWrapper.new(Aws::S3::Object.new(bucket_name,
 object_content))
  return unless wrapper.put_object_encrypted(object_content, encryption)

  puts "Put your content into #{bucket_name}:#{object_key} and encrypted it with
 #{encryption}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Contoh kode berikut menunjukkan cara menentukan status enkripsi objek yang ada.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectGetEncryptionWrapper
```

```
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Gets the object into memory.
  #
  # @return [Aws::S3::Types::GetObjectOutput, nil] The retrieved object data if
 successful; otherwise nil.
  def get_object
    @object.get
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't get object #{@object.key}. Here's why: #{e.message}"
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object.txt"

  wrapper = ObjectGetEncryptionWrapper.new(Aws::S3::Object.new(bucket_name,
 object_key))
  obj_data = wrapper.get_object
  return unless obj_data

  encryption = obj_data.server_side_encryption.nil? ? "no" :
 obj_data.server_side_encryption
  puts "Object #{object_key} uses #{encryption} encryption."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Jika enkripsi di sisi server tidak digunakan untuk objek yang disimpan di Amazon S3, metode tersebut akan menghasilkan `null`.

Untuk mengubah status enkripsi objek yang ada, buat salinan objek tersebut dan hapus objek sumber. Secara default, metode penyalinan tidak mengenkripsi target kecuali Anda secara eksplisit meminta enkripsi di sisi server. Anda dapat meminta enkripsi objek target dengan menentukan nilai `server_side_encryption` dalam argumen hash opsi, seperti yang

ditunjukkan dalam contoh kode Ruby berikut. Contoh kode mendemonstrasikan cara menyalin sebuah objek dan mengenkripsi salinan dengan SSE-S3.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectCopyEncryptWrapper
  attr_reader :source_object

  # @param source_object [Aws::S3::Object] An existing Amazon S3 object. This is
 used as the source object for
  #                                        copy actions.
  def initialize(source_object)
    @source_object = source_object
  end

  # Copy the source object to the specified target bucket, rename it with the target
 key, and encrypt it.
  #
  # @param target_bucket [Aws::S3::Bucket] An existing Amazon S3 bucket where the
 object is copied.
  # @param target_object_key [String] The key to give the copy of the object.
  # @return [Aws::S3::Object, nil] The copied object when successful; otherwise,
 nil.
  def copy_object(target_bucket, target_object_key, encryption)
    @source_object.copy_to(bucket: target_bucket.name, key: target_object_key,
 server_side_encryption: encryption)
    target_bucket.object(target_object_key)
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't copy #{@source_object.key} to #{target_object_key}. Here's why:
 #{e.message}"
  end
end

# Example usage:
def run_demo
  source_bucket_name = "doc-example-bucket1"
  source_key = "my-source-file.txt"
  target_bucket_name = "doc-example-bucket2"
  target_key = "my-target-file.txt"
  target_encryption = "AES256"

  source_bucket = Aws::S3::Bucket.new(source_bucket_name)
```

```
  wrapper = ObjectCopyEncryptWrapper.new(source_bucket.object(source_key))
  target_bucket = Aws::S3::Bucket.new(target_bucket_name)
  target_object = wrapper.copy_object(target_bucket, target_key, target_encryption)
  return unless target_object

  puts "Copied #{source_key} from #{source_bucket_name} to
 #{target_object.bucket_name}:#{target_object.key} and "\
       "encrypted the target with #{target_object.server_side_encryption}
 encryption."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Menggunakan AWS CLI

Untuk menentukan SSE-S3 saat Anda mengunggah objek dengan menggunakan AWS CLI, gunakan contoh berikut.

```
aws s3api put-object --bucket example-s3-bucket1 --key object-key-name --server-side-
encryption AES256  --body file path
```

Untuk informasi selengkapnya, lihat put-object dalam Referensi AWS CLI . Untuk menentukan SSE-S3 saat Anda menyalin objek dengan menggunakan AWS CLI, lihat copy-object.

Menggunakan AWS CloudFormation

Untuk contoh pengaturan enkripsi menggunakan AWS CloudFormation, lihat Membuat bucket dengan enkripsi default dan contoh Membuat bucket dengan menggunakan enkripsi AWS KMS sisi server dengan S3 Bucket Key dalam *`AWS::S3::Bucket ServerSideEncryptionRule`* topik di Panduan Pengguna.AWS CloudFormation

Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS)

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS

CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Enkripsi di sisi server adalah enkripsi data di tempat tujuan oleh aplikasi atau layanan yang menerimanya.

Amazon S3 secara otomatis mengaktifkan enkripsi di sisi server dengan kunci yang dikelola Amazon S3 (SSE-S3) untuk unggah objek baru.

Kecuali Anda menentukan sebaliknya, bucket menggunakan SSE-S3 secara default untuk mengenkripsi objek. Namun, Anda dapat memilih untuk mengonfigurasi bucket untuk menggunakan enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS) sebagai gantinya. Untuk informasi selengkapnya, lihat Menentukan enkripsi di sisi server dengan AWS KMS (SSE-KMS).

AWS KMS adalah layanan yang menggabungkan perangkat keras dan perangkat lunak yang aman dan sangat tersedia untuk menyediakan sistem manajemen kunci yang diskalakan untuk cloud. Amazon S3 menggunakan enkripsi sisi server dengan AWS KMS (SSE-KMS) untuk mengenkripsi data objek S3 Anda. Juga, ketika SSE-KMS diminta untuk objek, checksum S3 (sebagai bagian dari metadata objek) disimpan dalam bentuk terenkripsi. Untuk informasi selengkapnya tentang checksum, silakan lihat Memeriksa integritas objek.

Jika Anda menggunakan kunci KMS, Anda dapat menggunakan AWS KMS melalui AWS Management Consoleatau AWS KMS API untuk melakukan hal berikut:

- Membuat, melihat, mengedit, memantau, mengaktifkan atau menonaktifkan, memutar, dan menjadwalkan penghapusan kunci KMS secara terpusat.
- Menentukan kebijakan yang mengendalikan bagaimana dan oleh siapa kunci KMS digunakan.
- Audit penggunaannya untuk membuktikan bahwa mereka digunakan dengan benar. Audit yang didukung oleh API AWS KMS, tetapi tidak oleh AWS KMSAWS Management Console.

Kontrol keamanan AWS KMS dapat membantu Anda memenuhi persyaratan kepatuhan terkait enkripsi. Anda dapat menggunakan kunci KMS ini untuk melindungi data Anda di bucket Amazon S3. Saat Anda menggunakan enkripsi SSE-KMS dengan bucket S3, AWS KMS keys harus berada di Wilayah yang sama dengan bucket.

Ada biaya tambahan untuk penggunaan AWS KMS keys. Untuk informasi lebih umum, lihat konsep AWS KMS key dalam AWS Key Management Service Panduan Pengguna dan AWS KMS harga.

Izin

Untuk mengunggah objek yang dienkripsi dengan Amazon S3 AWS KMS key ke Amazon, Anda `kms:GenerateDataKey` memerlukan izin pada kunci. Untuk mengunduh objek yang dienkripsi dengan AWS KMS key, Anda memerlukan `kms:Decrypt` izin. Untuk informasi tentang AWS KMS izin yang diperlukan untuk unggahan multibagian, lihat. API dan izin unggahan multibagian

> **Important**
>
> Tinjau dengan cermat izin yang diberikan dalam kebijakan kunci KMS Anda. Selalu batasi izin kebijakan kunci KMS yang dikelola pelanggan hanya untuk kepala sekolah dan AWS layanan IAM yang harus mengakses tindakan kunci yang relevan. AWS KMS Untuk informasi selengkapnya, lihat Kebijakan utama di AWS KMS.

Topik

- AWS KMS keys
- Kunci Bucket Amazon S3
- Membutuhkan enkripsi di sisi server
- Konteks enkripsi
- Mengirim permintaan untuk objek AWS KMS terenkripsi
- Menentukan enkripsi di sisi server dengan AWS KMS (SSE-KMS)
- Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3

AWS KMS keys

Bila Anda menggunakan enkripsi sisi server dengan AWS KMS (SSE-KMS), Anda dapat menggunakan kunci AWS terkelola default, atau Anda dapat menentukan kunci terkelola pelanggan yang telah Anda buat. AWS KMS mendukung enkripsi amplop. S3 menggunakan AWS KMS fitur untuk enkripsi amplop untuk lebih melindungi data Anda. Enkripsi amplop adalah praktik mengenkripsi data plaintext Anda dengan kunci data, kemudian mengenkripsi kunci data tersebut dengan kunci KMS. Untuk informasi selengkapnya tentang enkripsi amplop, lihat Enkripsi amplop dalam Panduan Pengembang AWS Key Management Service .

Jika Anda tidak menentukan kunci terkelola pelanggan, Amazon S3 secara otomatis membuat saat pertama kali Anda Akun AWS menambahkan objek yang dienkripsi dengan SSE-KMS ke bucket. Kunci yang dikelola AWS Secara default, Amazon S3 menggunakan kunci KMS ini untuk SSE-KMS.

> **Note**
>
> Objek yang dienkripsi menggunakan SSE-KMS dengan Kunci yang dikelola AWS tidak dapat dibagikan lintas akun. Jika Anda perlu membagikan data lintas akun SSE-KMS, Anda harus menggunakan kunci yang dikelola pelanggan dari. AWS KMS

Jika Anda ingin menggunakan CMK untuk SSE-KMS, Anda dapat membuat CMK simetris sebelum Anda mengonfigurasi SSE-KMS. Kemudian, ketika Anda mengonfigurasi SSE-KMS untuk bucket Anda, Anda dapat menentukan CMK saat ini. Untuk informasi selengkapnya tentang kunci enkripsi simetris, lihat Kunci KMS enkripsi simetris dalam Panduan Pengembang AWS Key Management Service .

Membuat CMK memberikan Anda lebih banyak fleksibilitas dan kendali atas pelanggan. Misalnya, Anda dapat membuat, memutar, dan menonaktifkan CMK. Anda juga dapat menentukan kontrol akses dan mengaudit CMK yang Anda gunakan untuk melindungi data Anda. Untuk informasi selengkapnya tentang kunci yang AWS dikelola dan dikelola pelanggan, lihat Kunci dan AWS kunci pelanggan di Panduan AWS Key Management Service Pengembang.

> **Note**
>
> Saat Anda menggunakan enkripsi di sisi server dengan CMK yang disimpan di penyimpanan kunci eksternal, tidak seperti kunci KMS standar, Anda bertanggung jawab untuk memastikan ketersediaan dan ketahanan material kunci Anda. Untuk informasi selengkapnya tentang penyimpanan kunci eksternal dan cara mereka mengubah model tanggung jawab bersama, lihat Penyimpanan kunci eksternal di Panduan AWS Key Management Service Pengembang.

Menggunakan enkripsi SSE-KMS untuk operasi lintas akun

Saat menggunakan enkripsi untuk operasi lintas akun, perhatikan hal berikut:

- Jika Nama Sumber Daya AWS KMS key Amazon (ARN) atau alias tidak disediakan pada waktu permintaan atau melalui konfigurasi enkripsi default bucket, Kunci yang dikelola AWS (`aws/s3`) akan digunakan.

- Jika Anda mengunggah atau mengakses objek S3 dengan menggunakan prinsipal AWS Identity and Access Management (IAM) yang sama Akun AWS dengan kunci KMS Anda, Anda dapat menggunakan (). Kunci yang dikelola AWS `aws/s3`
- Jika Anda ingin memberikan akses lintas akun ke objek S3 Anda, gunakan CMK. Anda dapat mengonfigurasi kebijakan CMK untuk mengizinkan akses dari akun lain.
- Jika Anda menentukan kunci KMS yang dikelola pelanggan, kami sarankan untuk menggunakan ARN kunci KMS yang memenuhi syarat sepenuhnya. Jika Anda menggunakan alias kunci KMS sebagai gantinya, AWS KMS selesaikan kunci dalam akun pemohon. Perilaku ini dapat menghasilkan data yang dienkripsi dengan kunci KMS milik pemohon, dan bukan pemilik bucket.
- Anda harus menentukan kunci yang kepada Anda (pemohon) telah diberikan izin `Encrypt`. Untuk informasi selengkapnya, lihat Mengizinkan pengguna di akun lain untuk menggunakan kunci KMS untuk operasi kriptografi di Panduan Pengembang AWS Key Management Service .

Untuk informasi selengkapnya tentang kapan menggunakan kunci terkelola pelanggan dan kunci KMS AWS terkelola, lihat Haruskah saya menggunakan Kunci yang dikelola AWS atau kunci yang dikelola pelanggan untuk mengenkripsi objek saya di Amazon S3?

Alur kerja enkripsi SSE-KMS

Jika Anda memilih untuk mengenkripsi data menggunakan Kunci yang dikelola AWS atau kunci yang dikelola pelanggan, AWS KMS dan Amazon S3 melakukan tindakan enkripsi amplop berikut:

1. Amazon S3 meminta kunci data plaintext dan salinan kunci yang dienkripsi di bawah kunci KMS yang ditentukan.
2. AWS KMS menghasilkan kunci data, mengenkripsinya di bawah kunci KMS, dan mengirimkan kunci data teks biasa dan kunci data terenkripsi ke Amazon S3.
3. Amazon S3 mengenkripsi data menggunkaan kunci data dan menghapus kunci plaintext dari memori sesegera mungkin setelah digunakan.
4. Amazon S3 menyimpan kunci data terenkripsi sebagai metadata dengan data terenkripsi.

Saat Anda meminta agar data Anda didekripsi, Amazon S3 AWS KMS dan lakukan tindakan berikut:

1. Amazon S3 mengirimkan kunci data terenkripsi ke AWS KMS dalam permintaan. `Decrypt`
2. AWS KMS mendekripsi kunci data terenkripsi dengan menggunakan kunci KMS yang sama dan mengembalikan kunci data teks biasa ke Amazon S3.

3. Amazon S3 mendekripsi data terenkripsi, menggunakan kunci data teks biasa, dan menghapus kunci data teks biasa dari memori sesegera mungkin.

> **Important**
>
> Saat Anda menggunakan enkripsi sisi server AWS KMS key untuk Amazon S3, Anda harus memilih kunci KMS enkripsi simetris. Amazon S3 hanya mendukung kunci KMS enkripsi simetris. Untuk informasi selengkapnya terkait kunci ini, lihat Membuat kunci enkripsi simetris KMS dalam Panduan Pengembang AWS Key Management Service .

Mengaudit enkripsi SSE-KMS

Untuk mengidentifikasi permintaan yang menentukan SSE-KMS, Anda dapat menggunakan metrik Semua permintaan SSE-KMS dan % semua permintaan SSE-KMS dalam metrik Lensa Penyimpanan Amazon S3. Lensa Penyimpanan S3 adalah fitur analisis penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan objek. Anda juga dapat menggunakan jumlah bucket berkemampuan SSE-KMS dan bucket berkemampuan% SSE-KMS untuk memahami jumlah bucket yang (SSE-KMS) untuk enkripsi bucket default. Untuk informasi selengkapnya, lihat Menilai aktivitas penyimpanan dan penggunaan Anda dengan Lensa Penyimpanan S3. Untuk daftar lengkap metrik, lihat Glosarium metrik Lensa Penyimpanan S3.

Untuk mengaudit penggunaan AWS KMS kunci Anda untuk data terenkripsi SSE-KMS Anda, Anda dapat menggunakan log. AWS CloudTrail Anda bisa mendapatkan wawasan tentang operasi kriptografi Anda, seperti GenerateDataKeydan Decrypt. CloudTrail mendukung banyak nilai atribut untuk memfilter pencarian Anda, termasuk nama acara, nama pengguna, dan sumber acara.

Kunci Bucket Amazon S3

Saat mengonfigurasi enkripsi sisi server menggunakan AWS KMS (SSE-KMS), Anda dapat mengonfigurasi bucket untuk menggunakan S3 Bucket Keys untuk SSE-KMS. Menggunakan kunci tingkat ember untuk SSE-KMS dapat mengurangi biaya AWS KMS permintaan Anda hingga 99 persen dengan mengurangi lalu lintas permintaan dari Amazon S3 ke. AWS KMS

Ketika Anda mengonfigurasi bucket Anda untuk menggunakan Kunci Bucket S3 untuk SSE-KMS pada objek baru, AWS KMS menghasilkan kunci tingkat bucket yang digunakan untuk membuat kunci data untuk objek dalam bucket. Kunci Bucket S3 ini digunakan untuk jangka waktu terbatas dalam Amazon S3, semakin mengurangi kebutuhan Amazon S3 untuk membuat permintaan AWS

KMS untuk menyelesaikan operasi enkripsi. Untuk informasi lebih lanjut tentang menggunakan Kunci Bucket S3, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3.

Membutuhkan enkripsi di sisi server

Untuk mewajibkan enkripsi di sisi server terhadap semua objek dalam bucket Amazon S3 tertentu, Anda dapat menggunakan kebijakan bucket. Misalnya, kebijakan bucket berikut menolak izin objek unggah (`s3:PutObject`) untuk semua orang jika permintaan tidak menyertakan header `x-amz-server-side-encryption-aws-kms-key-id` yang meminta enkripsi di sisi server dengan SSE-KMS.

```
{
   "Version":"2012-10-17",
   "Id":"PutObjectPolicy",
   "Statement":[{
         "Sid":"DenyObjectsThatAreNotSSEKMS",
         "Effect":"Deny",
         "Principal":"*",
         "Action":"s3:PutObject",
         "Resource":"arn:aws:s3:::example-s3-bucket1/*",
         "Condition":{
            "Null":{
               "s3:x-amz-server-side-encryption-aws-kms-key-id":"true"
            }
         }
      }
   ]
}
```

Untuk mengharuskan seseorang AWS KMS key digunakan untuk mengenkripsi objek dalam ember, Anda dapat menggunakan kunci `s3:x-amz-server-side-encryption-aws-kms-key-id` kondisi. Untuk menentukan kunci KMS, Anda harus menggunakan kunci Amazon Resource Name (ARN) yang ada dalam `arn:aws:kms:`*`region`*`:`*`acct-id`*`:key/`*`key-id` format. AWS Identity and Access Management tidak memvalidasi jika string untuk `s3:x-amz-server-side-encryption-aws-kms-key-id` ada.

> **Note**
>
> Saat mengunggah sebuah objek, Anda dapat menentukan kunci KMS dengan menggunakan header `x-amz-server-side-encryption-aws-kms-key-id`. Jika header tidak ada

dalam permintaan, Amazon S3 menganggap bahwa Anda ingin menggunakan Kunci yang dikelola AWS. Terlepas dari itu, ID AWS KMS kunci yang digunakan Amazon S3 untuk enkripsi objek harus cocok dengan ID AWS KMS kunci dalam kebijakan, jika tidak Amazon S3 menolak permintaan tersebut.

Untuk daftar lengkap kunci kondisi khusus Amazon S3, lihat Kunci kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Konteks enkripsi

Konteks enkripsi adalah seperangkat pasangan nilai kunci yang berisi informasi kontekstual tambahan terkait data. Konteks enkripsi tidak dienkripsi. Ketika konteks enkripsi ditentukan untuk operasi enkripsi, Amazon S3 harus menentukan konteks enkripsi yang sama untuk operasi dekripsi. Jika tidak, dekripsi gagal. AWS KMS menggunakan konteks enkripsi sebagai data otentikasi tambahan (AAD) untuk mendukung enkripsi yang diautentikasi. Untuk informasi umum tentang konteks enkripsi, lihat Konteks enkripsi dalam Panduan Pengembang AWS Key Management Service .

Secara default, Amazon S3 menggunakan objek atau bucket Amazon Resource Name (ARN) sebagai pasangan konteks enkripsi:

- Jika Anda menggunakan SSE-KMS tanpa mengaktifkan Kunci Bucket S3, objek ARN digunakan sebagai konteks enkripsi.

  ```
  arn:aws:s3:::object_ARN
  ```

- Jika Anda menggunakan SSE-KMS dan mengaktifkan Kunci Bucket S3, ARN bucket akan digunakan sebagai konteks enkripsi. Untuk informasi selengkapnya tentang Kunci Bucket S3, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3.

  ```
  arn:aws:s3:::bucket_ARN
  ```

Anda secara opsional dapat memberikan pasangan konteks enkripsi tambahan dengan menggunakan `x-amz-server-side-encryption-context` header dalam permintaan s3:PutObject. Namun, karena konteks enkripsi tidak dienkripsi, pastikan bahwa itu tidak menyertakan informasi yang sensitif. Amazon S3 menyimpan pasangan kunci tambahan ini bersama dengan

konteks enkripsi default. Saat memproses permintaan PUT Anda, Amazon S3 menambahkan konteks enkripsi default ke `aws:s3:arn` yang Anda berikan.

Anda dapat menggunakan konteks enkripsi untuk mengidentifikasi dan mengategorikan operasi kriptografi Anda. Anda juga dapat menggunakan nilai ARN konteks enkripsi default untuk melacak permintaan yang relevan AWS CloudTrail dengan melihat Amazon S3 ARN mana yang digunakan dengan kunci enkripsi mana.

Di `requestParameters` bidang file CloudTrail log, konteks enkripsi terlihat mirip dengan yang berikut.

```
"encryptionContext": {
    "aws:s3:arn": "arn:aws:s3:::example-s3-bucket1/file_name"
}
```

Saat Anda menggunakan SSE-KMS dengan fitur S3 Bucket Keys opsional, nilai konteks enkripsi adalah ARN bucket.

```
"encryptionContext": {
    "aws:s3:arn": "arn:aws:s3:::example-s3-bucket1"
}
```

Mengirim permintaan untuk objek AWS KMS terenkripsi

> **Important**
>
> Semua `GET` dan `PUT` permintaan untuk objek AWS KMS terenkripsi harus dibuat menggunakan Secure Sockets Layer (SSL) atau Transport Layer Security (TLS). Permintaan juga harus ditandatangani menggunakan kredensyal yang valid, seperti AWS Signature Version 4 (atau AWS Signature Version 2).

AWS Signature Version 4 adalah proses menambahkan informasi otentikasi ke AWS permintaan yang dikirim oleh HTTP. Untuk keamanan, sebagian besar permintaan AWS harus ditandatangani dengan kunci akses, yang terdiri dari ID kunci akses dan kunci akses rahasia. Kedua kunci ini umumnya disebut sebagai kredensial keamanan Anda. Untuk informasi selengkapnya, lihat Permintaan Autentikasi (Tanda Tangan AWS Versi 4) dan Proses penandatanganan Tanda Tangan Versi 4.

> **Important**
>
> Jika objek Anda menggunakan SSE-KMS, jangan mengirim header permintaan enkripsi untuk permintaan `GET` dan permintaan `HEAD`. Jika tidak, Anda akan mendapatkan kesalahan HTTP 400 Bad Request.

Topik

- Menentukan enkripsi di sisi server dengan AWS KMS (SSE-KMS)
- Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3

Menentukan enkripsi di sisi server dengan AWS KMS (SSE-KMS)

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Semua bucket Amazon S3 memiliki enkripsi yang dikonfigurasi secara default, dan semua objek baru yang diunggah ke bucket S3 secara otomatis dienkripsi saat sedang tidak aktif. Enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) adalah konfigurasi enkripsi default untuk setiap bucket di Amazon S3. Untuk menggunakan jenis enkripsi yang berbeda, Anda dapat menentukan jenis enkripsi di sisi server yang akan digunakan dalam permintaan `PUT` S3, atau Anda dapat mengatur konfigurasi enkripsi default di bucket tujuan.

Jika Anda ingin menentukan jenis enkripsi yang berbeda dalam `PUT` permintaan Anda, Anda dapat menggunakan enkripsi sisi server dengan AWS Key Management Service () kunci (SSE-KMS AWS KMS), enkripsi sisi server dua lapis dengan kunci (DSSE-KMS), atau enkripsi sisi server dengan AWS KMS kunci yang disediakan pelanggan (SSE-C). Jika Anda ingin mengatur konfigurasi enkripsi default yang berbeda di dalam bucket tujuan, Anda dapat menggunakan SSE-KMS atau DSSE-KMS.

Anda dapat menerapkan enkripsi saat mengunggah objek baru, atau menyalin objek yang sudah ada.

Anda dapat menentukan SSE-KMS menggunakan konsol Amazon S3, operasi REST API, AWS SDK, dan (). AWS Command Line Interface AWS CLI Untuk informasi selengkapnya, lihat topik berikut.

> **Note**
>
> Anda dapat menggunakan Multi-wilayah AWS KMS keys di Amazon S3. Namun, Amazon S3 saat ini memperlakukan kunci multi-Wilayah selayaknya kunci satu Wilayah, dan tidak menggunakan fitur multi-Wilayah dari kunci tersebut. Untuk informasi selengkapnya, lihat Menggunakan kunci multi-Wilayah di Panduan Pengembang AWS Key Management Service .

> **Note**
>
> Jika Anda ingin menggunakan kunci KMS yang dimiliki oleh akun lain, Anda harus memiliki izin untuk menggunakan kunci tersebut. Untuk informasi selengkapnya tentang izin lintas akun untuk kunci KMS, lihat Membuat kunci KMS yang dapat digunakan oleh akun lain di Panduan Pengembang AWS Key Management Service .

Menggunakan konsol S3

Topik ini menjelaskan cara mengatur atau mengubah jenis enkripsi objek untuk menggunakan enkripsi sisi server dengan AWS Key Management Service (AWS KMS) kunci (SSE-KMS) dengan menggunakan konsol Amazon S3.

> **Note**
>
> - Jika Anda mengubah enkripsi objek, sebuah objek baru akan dibuat untuk menggantikan objek yang lama. Jika Penentuan Versi S3 diaktifkan, versi baru objek akan dibuat, dan objek yang sudah ada menjadi versi yang lebih lama. Peran yang mengubah properti juga menjadi pemilik objek baru (atau versi objek).
> - Jika Anda mengubah jenis enkripsi untuk objek yang memiliki tag yang ditentukan pengguna, Anda harus memiliki izin. `s3:GetObjectTagging` Jika Anda mengubah jenis

enkripsi untuk objek yang tidak memiliki tag yang ditentukan pengguna tetapi berukuran lebih dari 16 MB, Anda juga harus memiliki izin. `s3:GetObjectTagging`

Jika kebijakan bucket tujuan menolak `s3:GetObjectTagging` tindakan, jenis enkripsi untuk objek akan diperbarui, tetapi tag yang ditentukan pengguna akan dihapus dari objek, dan Anda akan menerima kesalahan.

Untuk menambahkan atau mengubah enkripsi untuk objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di dalam daftar Bucket, pilih nama bucket yang berisi objek.
4. Di dalam daftar Objek, pilih nama objek yang ingin Anda tambahkan atau ubah enkripsinya.

   Halaman detail objek muncul, dengan beberapa bagian yang menampilkan properti untuk objek Anda.
5. Pilih tab Properti.
6. Gulir ke bawah ke bagian Pengaturan enkripsi di sisi server, lalu pilih Edit.

   Halaman Edit enkripsi di sisi server terbuka.
7. Di bawah enkripsi di sisi server, untuk Pengaturan enkripsi, pilih Timpa pengaturan bucket enkripsi default.
8. Di bawah Jenis enkripsi, pilih Enkripsi sisi server dengan AWS Key Management Service kunci (SSE-KMS).

   > **Important**
   >
   > Jika Anda menggunakan opsi SSE-KMS untuk konfigurasi enkripsi default, Anda tunduk pada kuota permintaan per detik (RPS) AWS KMS. Untuk informasi selengkapnya tentang kuota AWS KMS dan cara meminta kenaikan kuota, lihat Kuota di Panduan Pengembang AWS Key Management Service .
9. Di bagian bawah AWS KMS kunci, lakukan salah satu langkah berikut untuk memilih kunci KMS Anda:

- Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari AWS KMS keys Anda, dan pilih Kunci KMS Anda dari daftar kunci yang tersedia.

  Kunci Kunci yang dikelola AWS (`aws/s3`) dan kunci terkelola pelanggan Anda muncul dalam daftar ini. Untuk informasi selengkapnya tentang CMK, lihat Kunci pelanggan dan AWS kunci di AWS Key Management Service Panduan Pengembang.
- Untuk memasukkan ARN kunci KMS, pilih Masukkan AWS KMS key ARN, lalu masukkan ARN kunci KMS Anda di bidang yang muncul.
- Untuk membuat kunci terkelola pelanggan baru di AWS KMS konsol, pilih Buat kunci KMS.

  Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat kunci di Panduan AWS Key Management Service Pengembang.

> **Important**
>
> Anda hanya dapat menggunakan tombol KMS yang tersedia Wilayah AWS sama dengan bucket. Konsol Amazon S3 hanya mencantumkan kunci 100 KMS pertama di Wilayah yang sama dengan bucket. Untuk menggunakan kunci KMS yang tidak terdaftar, Anda harus memasukkan ARN kunci KMS Anda. Jika Anda ingin menggunakan kunci KMS yang dimiliki oleh akun lain, Anda harus terlebih dahulu memiliki izin untuk menggunakan kunci tersebut, dan kemudian Anda harus memasukkan ARN kunci KMS.
> Amazon S3 hanya mendukung kunci KMS enkripsi simetris, dan tidak mendukung kunci KMS asimetris. Untuk informasi selengkapnya, lihat Mengidentifikasi tombol KMS simetris dan asimetris dalam Panduan Pengembang AWS Key Management Service .

10. Pilih Simpan perubahan.

> **Note**
>
> Tindakan ini menerapkan enkripsi untuk semua objek yang ditentukan. Saat mengenkripsi folder, tunggu hingga operasi penyimpanannya selesai sebelum menambahkan objek baru ke folder tersebut.

Penggunaan API REST

Saat Anda membuat sebuah objek—yaitu, ketika Anda mengunggah objek baru atau menyalin objek yang sudah ada—Anda dapat menentukan penggunaan enkripsi di sisi server dengan AWS KMS keys (SSE-KMS) untuk mengenkripsi data Anda. Untuk melakukannya, tambahkan header `x-amz-server-side-encryption` ke permintaan. Atur nilai header ke algoritma enkripsi. `aws:kms`. Amazon S3 mengonfirmasi bahwa objek Anda disimpan menggunakan SSE-KMS dengan mengembalikan header respons `x-amz-server-side-encryption`.

Jika Anda menentukan header `x-amz-server-side-encryption` dengan nilai `aws:kms`, Anda juga dapat menggunakan header permintaan berikut ini:

- `x-amz-server-side-encryption-aws-kms-key-id`
- `x-amz-server-side-encryption-context`
- `x-amz-server-side-encryption-bucket-key-enabled`

Topik

- Operasi API REST Amazon S3 yang mendukung SSE-KMS
- Konteks enkripsi (x-amz-server-side-encryption-context)
- AWS KMS ID kunci (x-amz-server-side-encryption-aws-kms-key-id)
- Tombol Bucket S3 () x-amz-server-side-encryption-aws-bucket-key-enabled

Operasi API REST Amazon S3 yang mendukung SSE-KMS

Operasi API REST berikut ini menerima header permintaan `x-amz-server-side-encryption`, `x-amz-server-side-encryption-aws-kms-key-id`, dan `x-amz-server-side-encryption-context`.

- PutObject–Saat Anda mengunggah data dengan menggunakan operasi API `PUT`, Anda dapat menentukan header permintaan ini.
- CopyObject–Saat Anda menyalin objek, Anda memiliki objek sumber dan objek target. Ketika Anda melewati header SSE-KMS dengan `CopyObject` operasi, mereka hanya diterapkan ke objek target. Saat Anda menyalin objek yang ada, terlepas dari apakah objek sumber dienkripsi atau tidak, objek tujuan tidak dienkripsi kecuali Anda secara eksplisit meminta enkripsi sisi server.
- POST Object— Saat Anda menggunakan `POST` operasi untuk mengunggah objek, alih-alih header permintaan, Anda memberikan informasi yang sama di bidang formulir.

- CreateMultipartUpload— Saat Anda mengunggah objek besar dengan menggunakan operasi API unggahan multibagian, Anda dapat menentukan header ini. Anda menentukan header ini dalam `CreateMultipartUpload` permintaan.

Header respons dari operasi API REST berikut mengembalikan header `x-amz-server-side-encryption` saat objek disimpan menggunakan enkripsi di sisi server.

- PutObject
- CopyObject
- POST Object
- CreateMultipartUpload
- UploadPart
- UploadPartCopy
- CompleteMultipartUpload
- GetObject
- HeadObject

> **Important**
>
> - Semua `GET` dan `PUT` permintaan untuk objek yang dilindungi oleh AWS KMS gagal jika Anda tidak membuat permintaan ini dengan menggunakan Secure Sockets Layer (SSL), Transport Layer Security (TLS), atau Signature Version 4.
> - Jika objek Anda menggunakan SSE-KMS, jangan mengirim header permintaan enkripsi untuk `GET` permintaan dan `HEAD` permintaan, atau Anda akan mendapatkan kesalahan HTTP 400. BadRequest

Konteks enkripsi (**`x-amz-server-side-encryption-context`**)

Jika Anda menentukan `x-amz-server-side-encryption:aws:kms`, API Amazon S3 mendukung konteks enkripsi dengan header `x-amz-server-side-encryption-context`. Konteks enkripsi adalah seperangkat pasangan nilai kunci yang berisi informasi kontekstual tambahan terkait data.

Amazon S3 secara otomatis menggunakan objek atau bucket Amazon Resource Name (ARN) sebagai pasangan konteks enkripsi. Jika Anda menggunakan SSE-KMS tanpa mengaktifkan Kunci Bucket S3, Anda menggunakan objek ARN sebagai konteks enkripsi Anda; misalnya, `arn:aws:s3:::object_ARN`. Namun, jika Anda menggunakan SSE-KMS dan mengaktifkan S3 Bucket Key, Anda menggunakan objek ARN sebagai konteks enkripsi Anda; misalnya, `arn:aws:s3:::bucket_ARN`.

Anda dapat secara opsional memberikan pasangan konteks enkripsi tambahan dengan menggunakan header `x-amz-server-side-encryption-context`. Namun, karena konteks enkripsi tidak dienkripsi, pastikan itu tidak menyertakan informasi sensitif. Amazon S3 menyimpan pasangan kunci tambahan ini bersama dengan konteks enkripsi default.

Untuk informasi tentang konteks enkripsi di Amazon S3, lihat Konteks enkripsi. Untuk informasi umum tentang konteks enkripsi, lihat AWS Key Management Service Konsep-Konteks enkripsi dalam Panduan Pengembang AWS Key Management Service .

AWS KMS ID kunci (**`x-amz-server-side-encryption-aws-kms-key-id`**)

Anda dapat menggunakan header `x-amz-server-side-encryption-aws-kms-key-id` untuk menentukan ID CMK, yang digunakan untuk melindungi data. Jika Anda menentukan header `x-amz-server-side-encryption:aws:kms` tetapi tidak memberikan header `x-amz-server-side-encryption-aws-kms-key-id`, Amazon S3 menggunakan Kunci yang dikelola AWS (`aws/s3`) untuk melindungi data tersebut. Jika Anda ingin menggunakan CMK, Anda harus memberikan header `x-amz-server-side-encryption-aws-kms-key-id` kunci yang dikelola pelanggan.

> **Important**
>
> Saat Anda menggunakan enkripsi sisi server AWS KMS key untuk Amazon S3, Anda harus memilih kunci KMS enkripsi simetris. Amazon S3 hanya mendukung kunci KMS enkripsi simetris. Untuk informasi selengkapnya terkait kunci ini, lihat Membuat kunci enkripsi simetris KMS dalam Panduan Pengembang AWS Key Management Service .

Tombol Bucket S3 () **`x-amz-server-side-encryption-aws-bucket-key-enabled`**

Anda dapat menggunakan header `x-amz-server-side-encryption-aws-bucket-key-enabled` permintaan untuk mengaktifkan atau menonaktifkan Kunci Bucket S3 di tingkat objek. S3 Bucket Keys mengurangi biaya AWS KMS permintaan Anda dengan mengurangi lalu lintas

permintaan dari Amazon S3 ke. AWS KMS Untuk informasi selengkapnya, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3.

Jika Anda menentukan header `x-amz-server-side-encryption:aws:kms` tapi tidak memberikan header `x-amz-server-side-encryption-aws-bucket-key-enabled`, objek Anda menggunakan pengaturan Kunci Bucket S3 pada bucket tujuan untuk mengenkripsi objek Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi Kunci Bucket S3 pada tingkat objek .

Menggunakan AWS CLI

Untuk menggunakan contoh AWS CLI perintah berikut, ganti *`user input placeholders`* dengan informasi Anda sendiri.

Saat Anda mengunggah objek baru atau menyalin objek yang ada, Anda dapat menentukan penggunaan enkripsi sisi server dengan AWS KMS kunci untuk mengenkripsi data Anda. Untuk melakukannya, tambahkan header `--server-side-encryption aws:kms` ke permintaan. Gunakan tombol `--ssekms-key-id` *`example-key-id`* untuk menambahkan AWS KMS kunci terkelola pelanggan yang Anda buat. Jika Anda menentukan`--server-side-encryption aws:kms`, tetapi tidak memberikan ID AWS KMS kunci, Amazon S3 akan menggunakan kunci AWS terkelola.

```
aws s3api put-object --bucket example-s3-bucket --key example-object-key --server-side-
encryption aws:kms --ssekms-key-id example-key-id --ssekms-encryption-context example-
encryption-context --body filepath
```

Anda dapat mengaktifkan atau menonaktifkan Kunci Bucket S3 pada `copy-object` operasi Anda `put-object` atau dengan menambahkan `--bucket-key-enabled` atau`--no-bucket-key-enabled`. Kunci Bucket S3 dapat mengurangi biaya AWS KMS permintaan Anda dengan mengurangi lalu lintas permintaan dari Amazon S3 ke. AWS KMS Untuk informasi selengkapnya, lihat Mengurangi biaya SSE-KMS dengan S3 Bucket Keys.

```
aws s3api put-object --bucket example-s3-bucket --key example-object-key --server-side-
encryption aws:kms --bucket-key-enabled --body filepath
```

Anda dapat menyalin objek dari bucket sumber ke bucket baru dan menentukan enkripsi SSE-KMS.

```
aws s3api copy-object --copy-source example-s3-bucket/example-object-key --
bucket example-s3-bucket2 --key example-object-key --server-side-encryption aws:kms --
sse-kms-key-id example-key-id --ssekms-encryption-context example-encryption-context
```

Menggunakan AWS SDK

Saat menggunakan AWS SDK, Anda dapat meminta Amazon S3 untuk AWS KMS keys digunakan untuk enkripsi sisi server. Contoh berikut menunjukkan bagaimana menggunakan SSE-KMS dengan AWS SDK untuk Java dan .NET. Untuk informasi tentang SDK lain, lihat Contoh kode dan pustaka di Pusat AWS Pengembang.

> **Important**
>
> Saat Anda menggunakan enkripsi sisi server AWS KMS key untuk Amazon S3, Anda harus memilih kunci KMS enkripsi simetris. Amazon S3 hanya mendukung kunci KMS enkripsi simetris. Untuk informasi selengkapnya terkait kunci ini, lihat Membuat kunci enkripsi simetris KMS dalam Panduan Pengembang AWS Key Management Service .

Operasi **`CopyObject`**

Saat menyalin objek, Anda menambahkan properti permintaan yang sama (`ServerSideEncryptionMethod` dan `ServerSideEncryptionKeyManagementServiceKeyId`) untuk meminta Amazon S3 menggunakan AWS KMS key. Untuk informasi selengkapnya tentang menyalin objek, lihat Menyalin, memindahkan, dan mengganti nama objek.

Operasi **`PUT`**

Java

Saat mengunggah objek menggunakan AWS SDK for Java, Anda dapat meminta Amazon S3 untuk menggunakan AWS KMS key dengan menambahkan properti seperti `SSEAwsKeyManagementParams` yang ditunjukkan dalam permintaan berikut:

```
PutObjectRequest putRequest = new PutObjectRequest(bucketName,
   keyName, file).withSSEAwsKeyManagementParams(new SSEAwsKeyManagementParams());
```

Dalam hal ini, Amazon S3 menggunakan Kunci yang dikelola AWS ()`aws/s3`. Untuk informasi selengkapnya, lihat Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS). Anda dapat secara opsional membuat kunci KMS enkripsi simetris dan menentukannya dalam permintaan, seperti yang ditunjukkan pada contoh berikut:

```
PutObjectRequest putRequest = new PutObjectRequest(bucketName,
```

```
    keyName, file).withSSEAwsKeyManagementParams(new
 SSEAwsKeyManagementParams(keyID));
```

Untuk informasi selengkapnya tentang membuat kunci terkelola pelanggan, lihat Memprogram AWS KMS API di Panduan AWS Key Management Service Pengembang.

Untuk contoh kode kerja mengunggah sebuah objek, lihat topik berikut ini. Untuk menggunakan contoh ini, Anda harus memperbarui contoh kode dan memberikan informasi enkripsi seperti yang ditunjukkan dalam fragmen kode sebelumnya.

- Untuk mengunggah objek dalam operasi tunggal, lihat Mengunggah Objek.
- Untuk unggahan multibagian yang menggunakan operasi API unggahan multibagian tingkat tinggi atau tingkat rendah, lihat. Pengunggahan objek menggunakan unggahan multibagian

.NET

Saat mengunggah objek menggunakan AWS SDK for .NET, Anda dapat meminta Amazon S3 untuk menggunakan AWS KMS key dengan menambahkan properti seperti `ServerSideEncryptionMethod` yang ditunjukkan dalam permintaan berikut:

```
PutObjectRequest putRequest = new PutObjectRequest
 {
     BucketName = example-s3-bucket,
     Key = keyName,
     // other properties
     ServerSideEncryptionMethod = ServerSideEncryptionMethod.AWSKMS
 };
```

Dalam hal ini, Amazon S3 menggunakan. Kunci yang dikelola AWS Untuk informasi selengkapnya, lihat Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS). Anda dapat secara opsional membuat kunci terkelola pelanggan enkripsi simetris Anda sendiri dan menentukannya dalam permintaan, seperti yang ditunjukkan pada contoh berikut:

```
PutObjectRequest putRequest1 = new PutObjectRequest
{
  BucketName = example-s3-bucket,
  Key = keyName,
  // other properties
  ServerSideEncryptionMethod = ServerSideEncryptionMethod.AWSKMS,
```

```
  ServerSideEncryptionKeyManagementServiceKeyId = keyId
};
```

Untuk informasi selengkapnya tentang membuat kunci terkelola pelanggan, lihat Memprogram AWS KMS API di Panduan AWS Key Management Service Pengembang.

Untuk contoh kode kerja mengunggah sebuah objek, lihat topik berikut ini. Untuk menggunakan contoh ini, Anda harus memperbarui contoh kode dan memberikan informasi enkripsi seperti yang ditunjukkan dalam fragmen kode sebelumnya.

- Untuk mengunggah objek dalam operasi tunggal, lihat Mengunggah Objek.
- Untuk unggahan multibagian yang menggunakan operasi API unggahan multibagian tingkat tinggi atau tingkat rendah, lihat. Pengunggahan objek menggunakan unggahan multibagian

URL yang ditandatangani sebelumnya

Java

Saat membuat URL presigned untuk objek yang dienkripsi dengan AWS KMS key, Anda harus secara eksplisit menentukan Signature Versi 4, seperti yang ditunjukkan pada contoh berikut:

```
ClientConfiguration clientConfiguration = new ClientConfiguration();
clientConfiguration.setSignerOverride("AWSS3V4SignerType");
AmazonS3Client s3client = new AmazonS3Client(
        new ProfileCredentialsProvider(), clientConfiguration);
...
```

Untuk contoh kode, lihat Berbagi objek dengan URL yang telah ditandatangani.

.NET

Saat membuat URL presigned untuk objek yang dienkripsi dengan AWS KMS key, Anda harus secara eksplisit menentukan Signature Versi 4, seperti yang ditunjukkan pada contoh berikut:

```
AWSConfigs.S3Config.UseSignatureVersion4 = true;
```

Untuk contoh kode, lihat Berbagi objek dengan URL yang telah ditandatangani.

Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3

Amazon S3 Bucket Keys mengurangi biaya enkripsi sisi server Amazon S3 AWS Key Management Service dengan AWS KMS kunci () (SSE-KMS). Menggunakan kunci tingkat ember untuk SSE-KMS dapat mengurangi biaya AWS KMS permintaan hingga 99 persen dengan mengurangi lalu lintas permintaan dari Amazon S3 ke. AWS KMS Dengan beberapa klik di AWS Management Console, dan tanpa perubahan apa pun pada aplikasi klien Anda, Anda dapat mengonfigurasi bucket Anda untuk menggunakan Kunci Bucket S3 untuk enkripsi SSE-KMS pada objek baru.

> **Note**
>
> Kunci Bucket S3 tidak didukung untuk enkripsi sisi server dua lapis dengan kunci AWS Key Management Service () (AWS KMS DSSE-KMS).

Kunci Bucket S3 untuk SSE-KMS

Beban kerja yang mengakses jutaan atau miliaran objek yang dienkripsi dengan SSE-KMS dapat menghasilkan volume permintaan yang besar. AWS KMSSaat Anda menggunakan SSE-KMS untuk melindungi data Anda tanpa Kunci Bucket S3, Amazon S3 menggunakan kunci data individual untuk setiap AWS KMS objek. Dalam hal ini, Amazon S3 melakukan panggilan ke AWS KMS setiap kali permintaan dibuat terhadap objek yang dienkripsi KMS. Untuk informasi tentang cara kerja SSE-KMS, lihat Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS).

Saat Anda mengonfigurasi bucket untuk menggunakan Kunci Bucket S3 untuk SSE-KMS, buat kunci tingkat AWS ember berumur pendek dari, lalu simpan sementara di S3. AWS KMS Kunci tingkat bucket ini akan membuat kunci data untuk objek baru selama siklus hidupnya. Kunci Bucket S3 digunakan untuk jangka waktu terbatas dalam Amazon S3, mengurangi kebutuhan S3 untuk membuat permintaan AWS KMS untuk menyelesaikan operasi enkripsi. Ini mengurangi lalu lintas dari S3 menjadi AWS KMS, memungkinkan Anda mengakses objek yang AWS KMS dienkripsi di Amazon S3 dengan biaya yang lebih murah dari biaya sebelumnya.

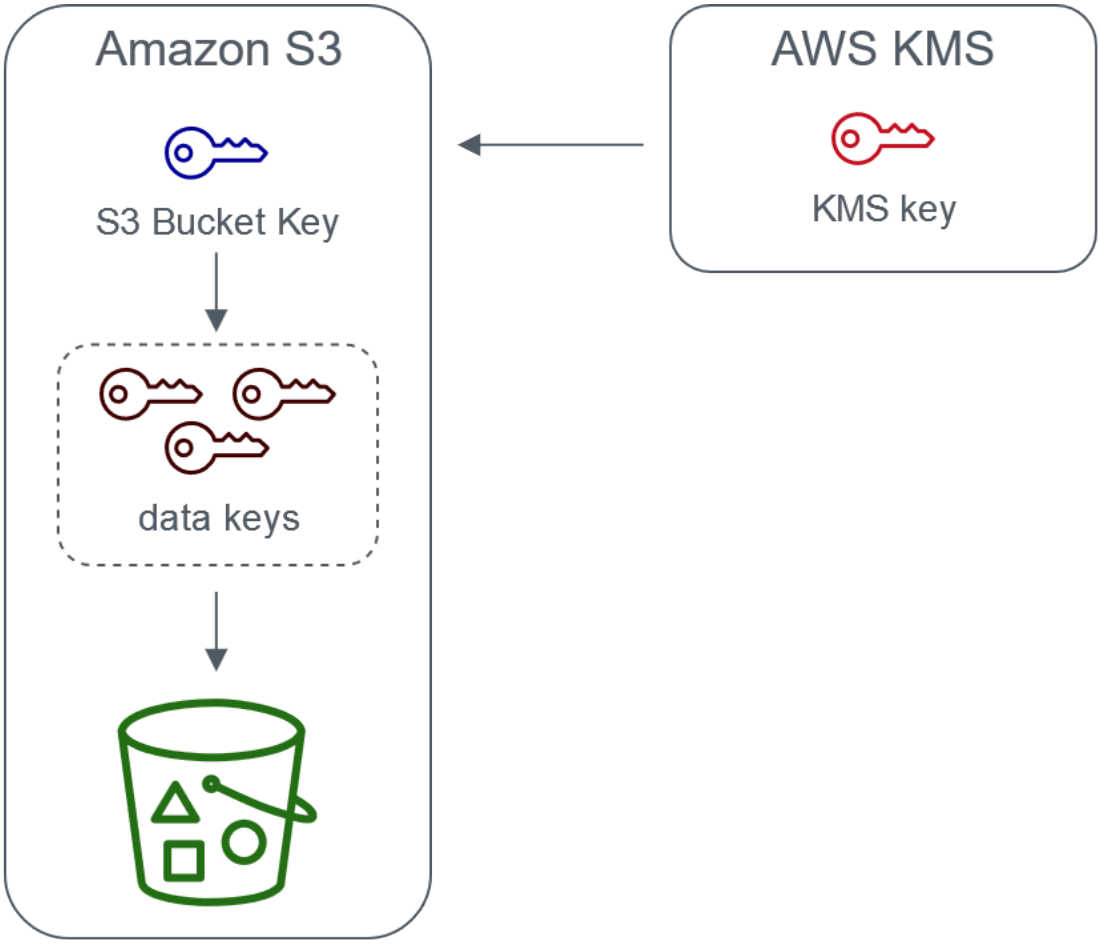
Kunci tingkat ember unik diambil setidaknya sekali per pemohon untuk memastikan bahwa akses pemohon ke kunci ditangkap dalam suatu peristiwa. AWS KMS CloudTrail Amazon S3 memperlakukan penelepon sebagai pemohon yang berbeda ketika mereka menggunakan peran atau akun yang berbeda, atau peran yang sama dengan kebijakan pelingkupan yang berbeda. AWS KMS penghematan permintaan mencerminkan jumlah pemohon, pola permintaan, dan usia relatif dari objek yang diminta. Misalnya, jumlah pemohon yang lebih sedikit, meminta beberapa objek

dalam jendela waktu terbatas, dan dienkripsi dengan kunci tingkat ember yang sama, menghasilkan penghematan yang lebih besar.

> **Note**
>
> Menggunakan S3 Bucket Keys memungkinkan Anda menghemat biaya AWS KMS permintaan dengan mengurangi permintaan AWS KMS untuk`Encrypt`,`GenerateDataKey`, dan `Decrypt` operasi melalui penggunaan kunci tingkat ember. Secara desain, permintaan berikutnya yang memanfaatkan kunci tingkat ember ini tidak menghasilkan permintaan AWS KMS API atau memvalidasi akses terhadap kebijakan kunci. AWS KMS

Bila Anda mengonfigurasi Kunci Bucket S3, objek yang sudah ada di bucket tidak menggunakan kunci Bucket S3. Untuk mengonfigurasi Kunci Bucket S3 untuk objek yang sudah ada, Anda dapat menggunakan operasi `CopyObject`. Untuk informasi selengkapnya, lihat Mengonfigurasi Kunci Bucket S3 pada tingkat objek .

Amazon S3 hanya akan berbagi Kunci Bucket S3 untuk objek dienkripsi oleh AWS KMS key yang sama. Kunci Bucket S3 kompatibel dengan kunci KMS yang dibuat oleh AWS KMS, bahan kunci impor, dan bahan kunci yang didukung oleh toko kunci khusus.



Server-side encryption with AWS Key Management service using an S3 Bucket Key

Mengonfigurasi Kunci Bucket S3

Anda dapat mengonfigurasi bucket untuk menggunakan Kunci Bucket S3 untuk SSE-KMS pada objek baru melalui konsol Amazon S3, AWS SDK, atau REST API. AWS CLI Dengan Kunci Bucket S3 diaktifkan di bucket Anda, objek yang diunggah dengan kunci SSE-KMS tertentu yang berbeda akan menggunakan Kunci Bucket S3 miliknya sendiri. Terlepas dari pengaturan Kunci Bucket S3 Anda, Anda dapat menyertakan header `x-amz-server-side-encryption-bucket-key-enabled` dengan nilai `true` atau `false` atau dalam permintaan Anda, untuk mengganti pengaturan bucket.

Sebelum mengonfigurasi bucket untuk menggunakan Kunci Bucket S3, tinjau Perubahan yang perlu diperhatikan sebelum mengaktifkan Kunci Bucket S3.

Mengonfigurasi Kunci Bucket S3 menggunakan konsol Amazon S3

Ketika Anda membuat bucket baru, Anda dapat mengonfigurasi bucket Anda untuk menggunakan Kunci Bucket S3 untuk SSE-KMS pada objek baru. Anda juga dapat mengonfigurasi bucket yang ada untuk menggunakan Kunci Bucket S3 untuk SSE-KMS pada objek baru dengan memperbarui properti bucket Anda.

Untuk informasi selengkapnya, lihat Mengonfigurasi bucket Anda untuk menggunakan Kunci Bucket S3 dengan SSE-KMS untuk objek baru.

REST API, AWS CLI, dan dukungan AWS SDK untuk S3 Bucket Keys

Anda dapat menggunakan REST API, AWS CLI, atau AWS SDK untuk mengonfigurasi bucket agar menggunakan Kunci Bucket S3 untuk SSE-KMS pada objek baru. Anda juga dapat mengaktifkan Kunci Bucket S3 pada tingkat objek.

Untuk informasi selengkapnya, lihat berikut ini:

- Mengonfigurasi Kunci Bucket S3 pada tingkat objek
- Mengonfigurasi bucket Anda untuk menggunakan Kunci Bucket S3 dengan SSE-KMS untuk objek baru

Operasi API berikut mendukung Kunci Bucket S3 untuk SSE-KMS:

- PutBucketEncryption
  - `ServerSideEncryptionRule` menerima parameter `BucketKeyEnabled` untuk mengaktifkan dan menonaktifkan Kunci Bucket S3.
- GetBucketEncryption
  - `ServerSideEncryptionRule` mengembalikan pengaturan untuk `BucketKeyEnabled`.
- PutObject, CopyObject, CreateMultipartUpload, dan POST Object
  - Anda dapat menggunakan header permintaan `x-amz-server-side-encryption-bucket-key-enabled` untuk mengaktifkan atau menonaktifkan Kunci Bucket S3 pada tingkat objek.
- HeadObject, GetObject, UploadPartCopy, UploadPart, dan CompleteMultipartUpload
  - Respons header `x-amz-server-side-encryption-bucket-key-enabled` menunjukkan jika Kunci Bucket S3 diaktifkan atau dinonaktifkan untuk sebuah objek.

Bekerja dengan AWS CloudFormation

Di AWS CloudFormation, `AWS::S3::Bucket` sumber daya menyertakan properti enkripsi yang disebut `BucketKeyEnabled` yang dapat Anda gunakan untuk mengaktifkan atau menonaktifkan Kunci Bucket S3.

Untuk informasi selengkapnya, lihat Menggunakan AWS CloudFormation.

Perubahan yang perlu diperhatikan sebelum mengaktifkan Kunci Bucket S3

Sebelum mengaktifkan Kunci Bucket S3, perhatikan perubahan terkait berikut ini:

IAM atau kebijakan AWS KMS utama

Jika kebijakan atau kebijakan AWS KMS utama AWS Identity and Access Management (IAM) yang ada menggunakan objek Amazon Resource Name (ARN) sebagai konteks enkripsi untuk mempersempit atau membatasi akses ke kunci KMS, kebijakan ini tidak akan berfungsi dengan Kunci Bucket S3. Kunci Bucket S3 menggunakan bucket ARN sebagai konteks enkripsi. Sebelum mengaktifkan Kunci Bucket S3, perbarui kebijakan IAM atau kebijakan AWS KMS utama untuk menggunakan ARN bucket sebagai konteks enkripsi.

Untuk informasi selengkapnya tentang konteks enkripsi dan Kunci Bucket S3, lihat Konteks enkripsi.

CloudTrail acara untuk AWS KMS

Setelah Anda mengaktifkan Kunci Bucket S3, AWS KMS CloudTrail peristiwa Anda mencatat ARN bucket Anda alih-alih ARN objek Anda. Selain itu, Anda melihat lebih sedikit CloudTrail peristiwa KMS untuk objek SSE-KMS di log Anda. Karena materi utama dibatasi waktu di Amazon S3, lebih sedikit permintaan yang dibuat. AWS KMS

Menggunakan kunci Bucket S3 dengan replikasi

Anda dapat menggunakan Kunci Bucket S3 dengan Replikasi Wilayah yang Sama (SRR) dan Replikasi Lintas-Wilayah (CRR).

Ketika Amazon S3 mereplikasi objek terenkripsi, umumnya mempertahankan pengaturan enkripsi objek replika di bucket tujuan. Namun, jika objek sumber tidak dienkripsi dan bucket tujuan Anda menggunakan enkripsi default atau Kunci Bucket S3, Amazon S3 mengenkripsi objek dengan konfigurasi bucket tujuan.

Contoh berikut menggambarkan bagaimana Kunci Bucket S3 bekerja dengan replikasi. Untuk informasi selengkapnya, lihat Mereplikasi objek terenkripsi (SSE-S3, SSE-KMS, SSE-KMS).

Example Contoh 1–Sumber objek menggunakan Kunci Bucket S3, bucket tujuan menggunakan enkripsi default

Jika objek sumber Anda menggunakan Kunci Bucket S3 namun bucket tujuan Anda menggunakan enkripsi default dengan SSE-KMS, objek replika akan mempertahankan pengaturan enkripsi Kunci Bucket S3 di bucket tujuan. Bucket tujuan masih menggunakan enkripsi default dengan SSE-KMS.

Example Contoh 2–Objek sumber tidak dienkripsi, bucket tujuan menggunakan Kunci Bucket S3 dengan SSE-KMS

Jika objek sumber Anda tidak dienkripsi dan bucket tujuan menggunakan Kunci Bucket S3 dengan SSE-KMS, objek replika dienkripsi menggunakan Kunci Bucket S3 dengan SSE-KMS di bucket tujuan. Ini menghasilkan objek sumber `ETag` yang berbeda dari objek replika `ETag`. Anda harus memperbarui aplikasi yang menggunakan `ETag` untuk mengakomodasi perbedaan ini.

Bekerja dengan Kunci Bucket S3

Untuk informasi lebih lanjut tentang mengaktifkan dan bekerja dengan Kunci Bucket S3, lihat bagian berikut:

- Mengonfigurasi bucket Anda untuk menggunakan Kunci Bucket S3 dengan SSE-KMS untuk objek baru
- Mengonfigurasi Kunci Bucket S3 pada tingkat objek
- Melihat pengaturan untuk Kunci Bucket S3

Mengonfigurasi bucket Anda untuk menggunakan Kunci Bucket S3 dengan SSE-KMS untuk objek baru

Saat mengonfigurasi enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS), Anda dapat mengonfigurasi bucket untuk menggunakan Kunci Bucket S3 untuk SSE-KMS pada objek baru. S3 Bucket Keys mengurangi lalu lintas permintaan dari Amazon S3 AWS KMS ke dan mengurangi biaya SSE-KMS. Untuk informasi selengkapnya, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3.

Anda dapat mengonfigurasi bucket untuk menggunakan Kunci Bucket S3 untuk SSE-KMS pada objek baru dengan menggunakan konsol Amazon S3, REST API, AWS SDK, (), atau. AWS Command Line Interface AWS CLI AWS CloudFormation Jika Anda ingin mengaktifkan atau nonaktifkan Kunci Bucket S3 untuk objek yang ada, Anda dapat menggunakan sebuah `CopyObject`

operasi. Untuk informasi selengkapnya, lihat Mengonfigurasi Kunci Bucket S3 pada tingkat objek dan Menggunakan Operasi Batch S3 untuk mengenkripsi objek dengan Kunci Bucket S3.

Saat Kunci Bucket S3 diaktifkan untuk bucket sumber atau tujuan, konteks enkripsi akan menjadi Amazon Resource Name (ARN) bucket, bukan ARN objek (misalnya, `arn:aws:s3:::bucket_ARN`). Anda harus memperbarui kebijakan IAM Anda untuk menggunakan ARN bucket untuk konteks enkripsi. Untuk informasi selengkapnya, lihat Replikasi dan Kunci Bucket S3.

Contoh berikut menggambarkan bagaimana Kunci Bucket S3 bekerja dengan replikasi. Untuk informasi selengkapnya, lihat Mereplikasi objek terenkripsi (SSE-S3, SSE-KMS, SSE-KMS).

Prasyarat

Sebelum mengonfigurasi bucket untuk menggunakan Kunci Bucket S3, tinjau Perubahan yang perlu diperhatikan sebelum mengaktifkan Kunci Bucket S3.

Menggunakan konsol S3

Di konsol S3, Anda dapat mengaktifkan atau nonaktifkan Kunci Bucket S3 untuk bucket baru atau yang sudah ada. Objek di konsol S3 mewarisi pengaturan Kunci Bucket S3 mereka dari konfigurasi bucket. Saat Anda mengaktifkan Kunci Bucket S3 untuk bucket Anda, objek baru yang Anda unggah ke bucket menggunakan Kunci Bucket S3 untuk SSE-KMS.

Mengunggah, menyalin, atau memodifikasi objek dalam bucket yang memiliki Kunci Bucket S3 diaktifkan

Jika Anda unggah, memodifikasi, atau menyalin objek dalam bucket yang mengaktifkan Kunci Bucket S3, pengaturan Kunci Bucket S3 untuk objek tersebut dapat diperbarui agar sesuai dengan konfigurasi bucket.

Jika sebuah objek telah mengaktifkan Kunci Bucket S3, pengaturan Kunci Bucket S3 untuk objek tersebut tidak berubah saat Anda menyalin atau memodifikasi objek. Namun, jika Anda mengubah atau menyalin objek yang tidak memiliki Kunci Bucket S3 yang diaktifkan, dan bucket tujuan memiliki konfigurasi Kunci Bucket S3, objek mewarisi pengaturan Kunci Bucket S3 bucket tujuan. Misalnya, jika objek sumber Anda tidak mengaktifkan Kunci Bucket S3 namun bucket tujuan mengaktifkan Kunci Bucket S3, Kunci Bucket S3 diaktifkan untuk objek tersebut.

Untuk mengaktifkan Kunci Bucket S3 saat Anda membuat bucket baru

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih Buat bucket.
4. Masukkan nama bucket Anda, dan pilih Wilayah AWS.
5. Di bawah Enkripsi default, untuk Jenis kunci enkripsi, pilih kunci AWS Key Management Service (SSE-KMS).
6. Di bagian bawah AWS KMS kunci, lakukan salah satu langkah berikut untuk memilih kunci KMS Anda:

   - Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari Anda AWS KMS keys, lalu pilih kunci KMS Anda dari daftar kunci yang tersedia.

     Kunci Kunci yang dikelola AWS (`aws/s3`) dan kunci terkelola pelanggan Anda muncul dalam daftar ini. Untuk informasi selengkapnya tentang kunci yang dikelola pelanggan, lihat Kunci dan AWS kunci pelanggan di Panduan AWS Key Management Service Pengembang.
   - Untuk memasukkan ARN kunci KMS, pilih Masukkan AWS KMS key ARN, dan masukkan ARN kunci KMS Anda di bidang yang muncul.
   - Untuk membuat kunci terkelola pelanggan baru di AWS KMS konsol, pilih Buat kunci KMS.

     Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat Kunci di Panduan AWS Key Management Service Pengembang.
7. Pada Kunci Bucket, pilih Aktifkan.
8. Pilih Buat bucket.

   Amazon S3 menciptakan bucket Anda dengan Kunci Bucket S3 yang diaktifkan. Objek baru yang Anda unggah ke bucket akan menggunakan Kunci Bucket S3.

   Untuk menonaktifkan Kunci Bucket S3, ikuti langkah-langkah sebelumnya, dan pilih Nonaktifkan.

Cara mengaktifkan Kunci Bucket S3 untuk bucket yang ada

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.

3. Di dalam daftar Bucket, pilih bucket yang ingin Anda aktifkan Kunci Bucket S3-nya.
4. Pilih tab Properti.
5. Di bagian bawah Enkripsi default, pilih Edit.
6. Di bawah Enkripsi default, untuk Jenis kunci enkripsi, pilih kunci AWS Key Management Service (SSE-KMS).
7. Di bagian bawah AWS KMS kunci, lakukan salah satu langkah berikut untuk memilih kunci KMS Anda:

   - Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari Anda AWS KMS keys, lalu pilih kunci KMS Anda dari daftar kunci yang tersedia.

     Kunci Kunci yang dikelola AWS (`aws/s3`) dan kunci terkelola pelanggan Anda muncul dalam daftar ini. Untuk informasi selengkapnya tentang kunci yang dikelola pelanggan, lihat Kunci dan AWS kunci pelanggan di Panduan AWS Key Management Service Pengembang.
   - Untuk memasukkan ARN kunci KMS, pilih Masukkan AWS KMS key ARN, dan masukkan ARN kunci KMS Anda di bidang yang muncul.
   - Untuk membuat kunci terkelola pelanggan baru di AWS KMS konsol, pilih Buat kunci KMS.

     Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat Kunci di Panduan AWS Key Management Service Pengembang.
8. Pada Kunci Bucket, pilih Aktifkan.
9. Pilih Simpan perubahan.

   Amazon S3 mengaktifkan Kunci Bucket S3 untuk objek baru yang ditambahkan ke bucket Anda. Objek yang ada tidak menggunakan Kunci Bucket S3. Untuk mengonfigurasi Kunci Bucket S3 untuk objek yang sudah ada, Anda dapat menggunakan operasi `CopyObject`. Untuk informasi selengkapnya, lihat Mengonfigurasi Kunci Bucket S3 pada tingkat objek .

   Untuk menonaktifkan Kunci Bucket S3, ikuti langkah-langkah sebelumnya, dan pilih Nonaktifkan.

Penggunaan API REST

Anda dapat menggunakan PutBucketEncryptionuntuk mengaktifkan atau menonaktifkan Kunci Bucket S3 untuk bucket Anda. Untuk mengonfigurasi Kunci Bucket S3 dengan`PutBucketEncryption`, gunakan tipe ServerSideEncryptionRuledata, yang mencakup enkripsi default dengan SSE-KMS. Anda juga dapat menggunakan CMK dengan menentukan ID kunci KMS untuk kunci yang dikelola pelanggan.

Untuk informasi selengkapnya dan contoh sintaks, lihat PutBucketEncryption.

Menggunakan AWS SDK for Java

Contoh berikut ini mengaktifkan enkripsi bucket default dengan SSE-KMS dan Kunci Bucket S3 dengan menggunakan AWS SDK for Java.

Java

```
AmazonS3 s3client = AmazonS3ClientBuilder.standard()
    .withRegion(Regions.DEFAULT_REGION)
    .build();

ServerSideEncryptionByDefault serverSideEncryptionByDefault = new
 ServerSideEncryptionByDefault()
    .withSSEAlgorithm(SSEAlgorithm.KMS);
ServerSideEncryptionRule rule = new ServerSideEncryptionRule()
    .withApplyServerSideEncryptionByDefault(serverSideEncryptionByDefault)
    .withBucketKeyEnabled(true);
ServerSideEncryptionConfiguration serverSideEncryptionConfiguration =
    new ServerSideEncryptionConfiguration().withRules(Collections.singleton(rule));

SetBucketEncryptionRequest setBucketEncryptionRequest = new
 SetBucketEncryptionRequest()
    .withServerSideEncryptionConfiguration(serverSideEncryptionConfiguration)
    .withBucketName(bucketName);

s3client.setBucketEncryption(setBucketEncryptionRequest);
```

Menggunakan AWS CLI

Contoh berikut ini mengaktifkan enkripsi bucket default dengan SSE-KMS dan Kunci Bucket S3 dengan menggunakan AWS CLI. Ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3api put-bucket-encryption --bucket example-s3-bucket --server-side-encryption-
configuration '{
        "Rules": [
            {
                "ApplyServerSideEncryptionByDefault": {
                    "SSEAlgorithm": "aws:kms",
                    "KMSMasterKeyID": "KMS-Key-ARN"
```

```
                },
                "BucketKeyEnabled": true
            }
        ]
    }'
```

Menggunakan AWS CloudFormation

Untuk informasi selengkapnya tentang mengonfigurasi Kunci Bucket S3 dengan AWS CloudFormation, lihat AWS::S3::Bucket ServerSideEncryptionRuledi AWS CloudFormation Panduan Pengguna.

Mengonfigurasi Kunci Bucket S3 pada tingkat objek

Saat Anda melakukan operasi PUT atau COPY menggunakan REST API, AWS SDK, atau AWS CLI, Anda dapat mengaktifkan atau menonaktifkan Kunci Bucket S3 di tingkat objek dengan menambahkan header `x-amz-server-side-encryption-bucket-key-enabled` permintaan dengan nilai `true` ataufalse. S3 Bucket Keys mengurangi biaya enkripsi sisi server menggunakan AWS Key Management Service (AWS KMS) (SSE-KMS) dengan mengurangi lalu lintas permintaan dari Amazon S3 ke. AWS KMS Untuk informasi selengkapnya, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3.

Saat Anda mengonfigurasi Kunci Bucket S3 untuk suatu objek menggunakan operasi PUT atau COPY, Amazon S3 hanya memperbarui pengaturan untuk objek tersebut. Pengaturan Kunci Bucket S3 untuk bucket tujuan tidak berubah. Jika Anda mengirimkan permintaan PUT atau COPY untuk objek terenkripsi KMS ke dalam bucket dengan S3 Bucket Keys diaktifkan, operasi tingkat objek Anda akan secara otomatis menggunakan S3 Bucket Keys, kecuali jika Anda menonaktifkan kunci di header permintaan. Jika Anda tidak menentukan Kunci Bucket S3 untuk objek Anda, Amazon S3 menerapkan pengaturan Kunci Bucket S3 untuk bucket tujuan ke objek.

Prasyarat:

Sebelum mengonfigurasi bucket untuk menggunakan Kunci Bucket S3, tinjau Perubahan yang perlu diperhatikan sebelum mengaktifkan Kunci Bucket S3.

Topik

- Operasi Batch Amazon S3
- Penggunaan API REST
- Menggunakan AWS SDK for Java PutObject ()
- Menggunakan AWS CLI (PutObject)

Operasi Batch Amazon S3

Untuk mengenkripsi objek Amazon S3 yang tidak terenkripsi, Anda dapat menggunakan Operasi Batch Amazon S3. Anda harus menyediakan Operasi Batch S3 dengan daftar objek untuk dioperasikan, dan Operasi Batch akan memanggil masing-masing API untuk melakukan operasi tertentu.

Anda juga dapat menggunakan Operasi Salin Operasi Batch S3 untuk menyalin objek tidak terenkripsi yang ada, dan menuliskannya kembali ke bucket yang sama sebagai objek terenkripsi. Satu tugas Operasi Batch dapat melakukan operasi tertentu pada miliaran objek yang berisi data sebesar eksabita. Untuk informasi selengkapnya, lihat Melakukan operasi batch berskala besar pada objek Amazon S3 dan Mengenkripsi objek dengan Operasi Batch Amazon S3.

Penggunaan API REST

Saat Anda menggunakan SSE-KMS, Anda dapat mengaktifkan Kunci Bucket S3 untuk sebuah objek menggunakan operasi API berikut ini:

- PutObject— Saat Anda mengunggah objek, Anda dapat menentukan header `x-amz-server-side-encryption-bucket-key-enabled` permintaan untuk mengaktifkan atau menonaktifkan Kunci Bucket S3 di tingkat objek.
- CopyObject— Saat Anda menyalin objek dan mengkonfigurasi SSE-KMS, Anda dapat menentukan header `x-amz-server-side-encryption-bucket-key-enabled` permintaan untuk mengaktifkan atau menonaktifkan Kunci Bucket S3 untuk objek Anda.
- Objek POST – Saat Anda menggunakan `POST` operasi untuk mengunggah objek dan mengonfigurasi SSE-KMS, Anda dapat menggunakan bidang formulir `x-amz-server-side-encryption-bucket-key-enabled` untuk mengaktifkan atau nonaktifkan Kunci Bucket S3 untuk objek Anda.
- CreateMultipartUpload— Saat Anda mengunggah objek besar dengan menggunakan operasi `CreateMultipartUpload` API dan mengonfigurasi SSE-KMS, Anda dapat menggunakan header `x-amz-server-side-encryption-bucket-key-enabled` permintaan untuk mengaktifkan atau menonaktifkan Kunci Bucket S3 untuk objek Anda.

Untuk mengaktifkan Kunci Bucket S3 pada tingkat objek, termasuk header permintaan `x-amz-server-side-encryption-bucket-key-enabled`. Untuk informasi selengkapnya tentang SSE-KMS dan API REST, lihat Penggunaan API REST.

Menggunakan AWS SDK for Java PutObject ()

Anda dapat menggunakan contoh berikut untuk mengonfigurasi Kunci Bucket S3 di tingkat objek menggunakan AWS SDK for Java.

Java

```
AmazonS3 s3client = AmazonS3ClientBuilder.standard()
    .withRegion(Regions.DEFAULT_REGION)
    .build();

String bucketName = "DOC-EXAMPLE-BUCKET1";
String keyName = "key name for object";
String contents = "file contents";

PutObjectRequest putObjectRequest = new PutObjectRequest(bucketName, keyName,
 contents)
    .withBucketKeyEnabled(true);

s3client.putObject(putObjectRequest);
```

Menggunakan AWS CLI (PutObject)

Anda dapat menggunakan AWS CLI contoh berikut untuk mengonfigurasi Kunci Bucket S3 di tingkat objek sebagai bagian dari `PutObject` permintaan.

```
aws s3api put-object --bucket example-s3-bucket --key object key name --server-side-
encryption aws:kms --bucket-key-enabled --body filepath
```

Melihat pengaturan untuk Kunci Bucket S3

Anda dapat melihat setelan untuk Kunci Bucket S3 di tingkat bucket atau objek dengan menggunakan konsol Amazon S3, REST API,AWS CLI() AWS Command Line Interface , AWS atau SDK.

S3 Bucket Keys mengurangi lalu lintas permintaan dari Amazon S3 AWS KMS ke dan mengurangi biaya penggunaan enkripsi sisi server (SSE-KMS). AWS Key Management Service Untuk informasi selengkapnya, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3.

Untuk melihat pengaturan Kunci Bucket S3 untuk bucket atau objek yang mewarisi pengaturan Kunci Bucket S3 dari konfigurasi bucket, Anda memerlukan izin untuk melakukan tindakan `s3:GetEncryptionConfiguration`. Untuk informasi selengkapnya, lihat GetBucketEncryptiondi Referensi API Amazon Simple Storage Service.

Menggunakan konsol S3

Di konsol S3, Anda dapat melihat pengaturan Kunci Bucket S3 untuk bucket atau objek Anda. Pengaturan Kunci Bucket S3 diwarisi dari konfigurasi bucket kecuali objek sumber sudah memiliki Kunci Bucket S3 dikonfigurasi.

Objek dan folder dalam bucket yang sama dapat memiliki pengaturan Kunci Bucket S3 yang berbeda. Misalnya, jika Anda unggah objek menggunakan API REST dan mengaktifkan Kunci Bucket S3 untuk objek, objek mempertahankan pengaturan Kunci Bucket S3 di bucket tujuan, bahkan jika Kunci Bucket S3 dinonaktifkan di bucket tujuan. Sebagai contoh lain, jika Anda mengaktifkan kunci Bucket S3 untuk bucket yang ada, objek yang sudah ada di dalam bucket tidak menggunakan Kunci Bucket S3. Namun, objek baru memiliki kunci bucket S3 yang diaktifkan.

Untuk melihat pengaturan Kunci Bucket S3 untuk bucket Anda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di dalam daftar Bucket, pilih bucket yang ingin Anda aktifkan Kunci Bucket S3-nya.
4. Pilih Properti.
5. Di bagian Enkripsi default, di bawah Kunci Bucket, Anda melihat pengaturan Tombol Bucket S3 untuk bucket Anda.

   Jika Anda tidak dapat melihat pengaturan Kunci Bucket S3, Anda mungkin tidak memiliki izin untuk menjalankan tindakan `s3:GetEncryptionConfiguration`. Untuk informasi selengkapnya, lihat GetBucketEncryptiondi Referensi API Amazon Simple Storage Service.

Untuk melihat pengaturan Kunci Bucket S3 untuk bucket Anda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Bucket, pilih bucket yang ingin Anda aktifkan Kunci Bucket S3-nya.
3. Di daftar Objek, pilih nama objek Anda.

4. Di atas tab Detail, di bawah Pengaturan enkripsi di sisi server, memilih Edit.

   Pada Kunci Bucket, Anda melihat pengaturan Kunci Bucket S3 untuk objek Anda. Anda tidak bisa mengedit pengaturan ini.

Menggunakan AWS CLI

Untuk mengembalikan pengaturan Kunci Bucket S3 tingkat bucket

Untuk menggunakan contoh ini, ganti masing-masing *user input placeholder* dengan informasi Anda sendiri.

```
aws s3api get-bucket-encryption --bucket example-s3-bucket1
```

Untuk informasi selengkapnya, lihat get-bucket-encryptiondi Referensi AWS CLI Perintah.

Untuk mengembalikan pengaturan Kunci Bucket S3 tingkat objek

Untuk menggunakan contoh ini, ganti masing-masing *user input placeholder* dengan informasi Anda sendiri.

```
aws s3api head-object --bucket example-s3-bucket1 --key my_images.tar.bz2
```

Untuk informasi selengkapnya, lihat head-object dalam AWS CLI Referensi Perintah.

Penggunaan API REST

Untuk mengembalikan pengaturan Kunci Bucket S3 tingkat bucket

Untuk mengembalikan informasi enkripsi untuk bucket, termasuk pengaturan untuk Kunci Bucket S3, gunakan operasi `GetBucketEncryption`. Pengaturan Kunci bucket S3 dikembalikan di badan respons di elemen `ServerSideEncryptionConfiguration` dengan pengaturan `BucketKeyEnabled`. Untuk informasi selengkapnya, lihat GetBucketEncryptiondi Referensi API Amazon S3.

Untuk mengembalikan pengaturan tingkat objek untuk Kunci Bucket S3

Untuk mengembalikan status Kunci Bucket S3 untuk objek, gunakan operasi `HeadObject`. `HeadObject` mengembalikan header respons `x-amz-server-side-encryption-bucket-key-enabled` untuk menunjukkan apakah Kunci Bucket S3 diaktifkan atau dinonaktifkan untuk objek. Untuk informasi selengkapnya, lihat HeadObjectdi Referensi API Amazon S3.

Operasi API berikut juga mengembalikan header respons `x-amz-server-side-encryption-bucket-key-enabled` jika Kunci Bucket S3 dikonfigurasi untuk sebuah objek:

- PutObject
- PostObject
- CopyObject
- CreateMultipartUpload
- UploadPartCopy
- UploadPart
- CompleteMultipartUpload
- GetObject

Menggunakan enkripsi sisi server dual-layer dengan kunci (DSSE-KMS) AWS KMS

Menggunakan enkripsi sisi server dua lapis dengan kunci AWS Key Management Service (AWS KMS) (DSSE-KMS) menerapkan dua lapisan enkripsi ke objek saat diunggah ke Amazon S3. DSSE-KMS membantu Anda agar lebih mudah memenuhi standar kepatuhan yang mengharuskan Anda untuk menerapkan enkripsi multilayer ke data Anda dan memiliki kontrol penuh atas kunci enkripsi Anda.

Saat Anda menggunakan DSSE-KMS dengan bucket Amazon S3, AWS KMS kunci harus berada di Wilayah yang sama dengan bucket. Selain itu, ketika DSSE-KMS diminta untuk objek tersebut, checksum S3 yang merupakan bagian dari metadata objek disimpan dalam bentuk terenkripsi. Untuk informasi selengkapnya tentang checksum, lihat Memeriksa integritas objek.

Ada biaya tambahan untuk menggunakan DSSE-KMS dan. AWS KMS keys Untuk informasi lebih lanjut tentang harga DSSE-KMS, lihat AWS KMS key konsep dalam AWS Key Management Service Panduan Pengguna dan AWS KMS harga.

> **Note**
>
> Kunci Bucket S3 tidak didukung untuk DSSE-KMS.

Memerlukan enkripsi sisi server dua lapis dengan (DSSE-KMS) AWS KMS keys

Untuk mewajibkan enkripsi di sisi server terhadap semua objek dalam bucket Amazon S3 tertentu, Anda dapat menggunakan kebijakan bucket. Misalnya, kebijakan bucket berikut menolak izin objek

unggah (`s3:PutObject`) untuk semua orang jika permintaan tidak menyertakan header `x-amz-server-side-encryption` yang meminta enkripsi di sisi server dengan SSE-KMS.

```
{
            "Version":"2012-10-17",
            "Id":"PutObjectPolicy",
            "Statement":[{
                  "Sid":"DenyUnEncryptedObjectUploads",
                  "Effect":"Deny",
                  "Principal":"*",
                  "Action":"s3:PutObject",
                  "Resource":"arn:aws:s3:::example-s3-bucket1/*",
                  "Condition":{
                     "StringNotEquals":{
                        "s3:x-amz-server-side-encryption":"aws:kms:dsse"
                     }
                  }
               }
            ]
         }
```

Topik

- Menentukan enkripsi di sisi server dua lapis dengan kunci AWS KMS (DSSE-KMS)

Menentukan enkripsi di sisi server dua lapis dengan kunci AWS KMS (DSSE-KMS)

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Semua bucket Amazon S3 memiliki enkripsi yang dikonfigurasi secara default, dan semua objek baru yang diunggah ke bucket S3 secara otomatis dienkripsi saat sedang tidak aktif. Enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) adalah konfigurasi enkripsi default untuk setiap bucket di Amazon S3. Untuk menggunakan jenis enkripsi yang berbeda, Anda dapat menentukan jenis enkripsi di sisi server yang akan digunakan dalam permintaan PUT S3, atau Anda dapat mengatur konfigurasi enkripsi default di bucket tujuan.

Jika Anda ingin menentukan jenis enkripsi yang berbeda dalam PUT permintaan Anda, Anda dapat menggunakan enkripsi sisi server dengan AWS Key Management Service () kunci (SSE-KMS AWS KMS), enkripsi sisi server dua lapis dengan kunci (DSSE-KMS), atau enkripsi sisi server dengan AWS KMS kunci yang disediakan pelanggan (SSE-C). Jika Anda ingin mengatur konfigurasi enkripsi default yang berbeda di dalam bucket tujuan, Anda dapat menggunakan SSE-KMS atau DSSE-KMS.

Anda dapat menerapkan enkripsi saat mengunggah objek baru, atau menyalin objek yang sudah ada.

Anda dapat menentukan DSSE-KMS menggunakan konsol Amazon S3, Amazon S3 API REST, dan AWS Command Line Interface (AWS CLI). Untuk informasi selengkapnya, lihat topik berikut.

> **Note**
>
> Anda dapat menggunakan Multi-wilayah AWS KMS keys di Amazon S3. Namun, Amazon S3 saat ini memperlakukan kunci multi-Wilayah selayaknya kunci satu Wilayah, dan tidak menggunakan fitur multi-Wilayah dari kunci tersebut. Untuk informasi selengkapnya, lihat Menggunakan kunci multi-Wilayah di Panduan Pengembang AWS Key Management Service .

> **Note**
>
> Jika Anda ingin menggunakan kunci KMS yang dimiliki oleh akun lain, Anda harus memiliki izin untuk menggunakan kunci tersebut. Untuk informasi selengkapnya tentang izin lintas akun untuk kunci KMS, lihat Membuat kunci KMS yang dapat digunakan oleh akun lain di Panduan Pengembang AWS Key Management Service .

Menggunakan konsol S3

Bagian ini menjelaskan cara mengatur atau mengubah jenis enkripsi objek untuk menggunakan enkripsi sisi server dua lapis dengan AWS Key Management Service (AWS KMS) kunci (DSSE-KMS) dengan menggunakan konsol Amazon S3.

> **Note**
>
> - Jika Anda mengubah metode enkripsi objek, objek baru akan dibuat untuk menggantikan objek lama. Jika Penentuan Versi S3 diaktifkan, versi baru objek akan dibuat, dan objek yang sudah ada menjadi versi yang lebih lama. Peran yang mengubah properti juga menjadi pemilik objek baru (atau versi objek).
> - Jika Anda mengubah jenis enkripsi untuk objek yang memiliki tag yang ditentukan pengguna, Anda harus memiliki izin. `s3:GetObjectTagging` Jika Anda mengubah jenis enkripsi untuk objek yang tidak memiliki tag yang ditentukan pengguna tetapi berukuran lebih dari 16 MB, Anda juga harus memiliki izin. `s3:GetObjectTagging`
>
> Jika kebijakan bucket tujuan menolak `s3:GetObjectTagging` tindakan, jenis enkripsi untuk objek akan diperbarui, tetapi tag yang ditentukan pengguna akan dihapus dari objek, dan Anda akan menerima kesalahan.

Untuk menambahkan atau mengubah enkripsi untuk objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di dalam daftar Bucket, pilih nama bucket yang berisi objek yang ingin Anda enkripsi.
4. Di dalam daftar Objek, pilih nama objek yang ingin Anda tambahkan atau ubah enkripsinya.

   Halaman detail objek muncul, dengan beberapa bagian yang menampilkan properti untuk objek Anda.
5. Pilih tab Properti.
6. Gulir ke bawah ke bagian Enkripsi default, lalu pilih Edit.

   Halaman Edit enkripsi di sisi server terbuka.

7. Di bawah Jenis enkripsi, pilih Enkripsi sisi server dual-layer dengan AWS Key Management Service kunci (DSSE-KMS).
8. Di bagian bawah AWS KMS kunci, lakukan salah satu langkah berikut untuk memilih kunci KMS Anda:

   - Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari AWS KMS keys Anda, dan pilih Kunci KMS Anda dari daftar kunci yang tersedia.

     Kunci Kunci yang dikelola AWS (`aws/s3`) dan kunci terkelola pelanggan Anda muncul dalam daftar ini. Untuk informasi selengkapnya tentang CMK, lihat Kunci pelanggan dan AWS kunci di AWS Key Management Service Panduan Pengembang.
   - Untuk memasukkan ARN kunci KMS, pilih Masukkan AWS KMS key ARN, lalu masukkan ARN kunci KMS Anda di bidang yang muncul.
   - Untuk membuat kunci terkelola pelanggan baru di AWS KMS konsol, pilih Buat kunci KMS.

     Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat kunci di Panduan AWS Key Management Service Pengembang.

   > **Important**
   >
   > Anda hanya dapat menggunakan kunci KMS yang tersedia di Wilayah AWS yang sama dengan bucket. Konsol Amazon S3 hanya mencantumkan kunci 100 KMS pertama di Wilayah yang sama dengan bucket. Untuk menggunakan kunci KMS yang tidak terdaftar, Anda harus memasukkan ARN kunci KMS Anda. Jika Anda ingin menggunakan kunci KMS yang dimiliki oleh akun yang berbeda, Anda harus terlebih dahulu memiliki izin untuk menggunakan kunci tersebut, dan kemudian Anda harus memasukkan kunci KMS ARN.
   > Amazon S3 hanya mendukung kunci KMS enkripsi simetris, dan tidak mendukung kunci KMS asimetris. Untuk informasi selengkapnya, lihat Mengidentifikasi kunci KMS asimetris dalam Panduan Pengembang AWS Key Management Service .

9. Untuk Kunci Bucket, pilih Nonaktifkan. Kunci Bucket S3 tidak didukung untuk DSSE-KMS.
10. Pilih Simpan perubahan.

> **Note**
>
> Tindakan ini menerapkan enkripsi untuk semua objek yang ditentukan. Saat mengenkripsi folder, tunggu hingga operasi penyimpanannya selesai sebelum menambahkan objek baru ke folder tersebut.

Penggunaan API REST

Saat Anda membuat objek — yaitu, ketika Anda mengunggah objek baru atau menyalin objek yang ada — Anda dapat menentukan penggunaan enkripsi sisi server dua lapis dengan (DSSE-KMS) untuk mengenkripsi data Anda. AWS KMS keys Untuk melakukannya, tambahkan header `x-amz-server-side-encryption` ke permintaan. Atur nilai header ke algoritma enkripsi. `aws:kms:dsse`. Amazon S3 mengonfirmasi bahwa objek Anda disimpan dengan enkripsi DSSE-KMS dengan mengembalikan header respons `x-amz-server-side-encryption`.

Jika Anda menentukan header `x-amz-server-side-encryption` dengan nilai `aws:kms:dsse`, Anda juga dapat menggunakan header permintaan berikut ini:

- `x-amz-server-side-encryption-aws-kms-key-id: SSEKMSKeyId`
- `x-amz-server-side-encryption-context: SSEKMSEncryptionContext`

Topik

- Operasi API REST Amazon S3 yang mendukung DSSE-KMS
- Konteks enkripsi (x-amz-server-side-encryption-context)
- AWS KMS ID kunci (x-amz-server-side-encryption-aws-kms-key-id)

Operasi API REST Amazon S3 yang mendukung DSSE-KMS

Operasi API REST berikut ini menerima header permintaan `x-amz-server-side-encryption`, `x-amz-server-side-encryption-aws-kms-key-id`, dan `x-amz-server-side-encryption-context`.

- PutObject–Saat Anda mengunggah data dengan menggunakan operasi API PUT, Anda dapat menentukan header permintaan ini.
- CopyObject–Saat Anda menyalin objek, Anda memiliki objek sumber dan objek target. Saat Anda melewati header DSSE-KMS dengan operasi `CopyObject`, itu diterapkan hanya untuk objek

target. Saat Anda menyalin objek yang ada, terlepas dari apakah objek sumbernya dienkripsi atau tidak, objek tujuan tidak dienkripsi kecuali jika Anda secara eksplisit meminta enkripsi di sisi server.

- POST Object–Saat Anda menggunakan operasi `POST` untuk mengunggah objek, alih-alih header permintaan, Anda memberikan informasi yang sama di kolom formulir.
- CreateMultipartUpload–Saat Anda mengunggah objek besar dengan menggunakan unggahan multibagian, Anda dapat menentukan header ini dalam permintaan `CreateMultipartUpload`.

Header respons dari operasi API REST berikut mengembalikan header `x-amz-server-side-encryption` saat objek disimpan dengan enkripsi di sisi server.

- PutObject
- CopyObject
- POST Objek
- CreateMultipartUpload
- UploadPart
- UploadPartCopy
- CompleteMultipartUpload
- GetObject
- HeadObject

> **Important**
>
> - Semua `GET` dan `PUT` permintaan untuk objek yang dilindungi oleh AWS KMS gagal jika Anda tidak membuatnya dengan menggunakan Secure Sockets Layer (SSL), Transport Layer Security (TLS), atau Signature Version 4.
> - Jika objek Anda menggunakan DSSE-KMS, jangan mengirim header permintaan enkripsi untuk permintaan `GET` dan permintaan `HEAD`, atau Anda akan mendapatkan kesalahan HTTP 400 (Bad Request).

Konteks enkripsi (**`x-amz-server-side-encryption-context`**)

Jika Anda menentukan `x-amz-server-side-encryption:aws:kms:dsse`, API Amazon S3 mendukung konteks enkripsi dengan header `x-amz-server-side-encryption-context`.

Konteks enkripsi adalah seperangkat pasangan nilai kunci yang berisi informasi kontekstual tambahan terkait data.

Amazon S3 secara otomatis menggunakan Amazon Resource Name (ARN) objek sebagai pasangan konteks enkripsi; misalnya, `arn:aws:s3:::object_ARN`.

Anda dapat secara opsional memberikan pasangan konteks enkripsi tambahan dengan menggunakan header `x-amz-server-side-encryption-context`. Namun, karena konteks enkripsi tidak dienkripsi, pastikan bahwa itu tidak menyertakan informasi yang sensitif. Amazon S3 menyimpan pasangan kunci tambahan ini bersama dengan konteks enkripsi default.

Untuk informasi tentang konteks enkripsi di Amazon S3, lihat Konteks enkripsi. Untuk informasi umum tentang konteks enkripsi, lihat AWS Key Management Service Konsep-Konteks enkripsi dalam Panduan Pengembang AWS Key Management Service .

AWS KMS ID kunci (**`x-amz-server-side-encryption-aws-kms-key-id`**)

Anda dapat menggunakan header `x-amz-server-side-encryption-aws-kms-key-id` untuk menentukan ID CMK, yang digunakan untuk melindungi data. Jika Anda menentukan `x-amz-server-side-encryption:aws:kms:dsse` header tetapi tidak memberikan `x-amz-server-side-encryption-aws-kms-key-id` header, Amazon S3 menggunakan Kunci yang dikelola AWS (`aws/s3`) untuk melindungi data. Jika Anda ingin menggunakan CMK, Anda harus memberikan header `x-amz-server-side-encryption-aws-kms-key-id` kunci yang dikelola pelanggan.

> **Important**
>
> Saat Anda menggunakan enkripsi sisi server AWS KMS key untuk Amazon S3, Anda harus memilih kunci KMS enkripsi simetris. Amazon S3 hanya mendukung kunci KMS enkripsi simetris. Untuk informasi selengkapnya terkait kunci ini, lihat Membuat kunci enkripsi simetris KMS dalam Panduan Pengembang AWS Key Management Service .

Menggunakan AWS CLI

Saat Anda mengunggah objek baru atau menyalin objek yang ada, Anda dapat menentukan penggunaan DSSE-KMS untuk mengenkripsi data Anda. Untuk melakukannya, tambahkan parameter `--server-side-encryption aws:kms:dsse` ke permintaan. Gunakan parameter `--ssekms-key-id example-key-id` untuk menambahkan kunci AWS KMS yang dikelola pelanggan yang telah Anda buat. Jika Anda menentukan`--server-side-encryption`

`aws:kms:dsse`, tetapi tidak memberikan ID AWS KMS kunci, maka Amazon S3 akan menggunakan kunci AWS terkelola ()`aws/s3`.

```
aws s3api put-object --bucket DOC-EXAMPLE-BUCKET --key example-object-key --server-
side-encryption aws:kms:dsse --ssekms-key-id example-key-id --body filepath
```

Anda dapat mengenkripsi objek yang tidak terenkripsi untuk menggunakan DSSE-KMS dengan menyalin objek kembali ke tempatnya.

```
aws s3api copy-object --bucket DOC-EXAMPLE-BUCKET --key example-object-key --
body filepath --bucket DOC-EXAMPLE-BUCKET --key example-object-key --sse aws:kms:dsse
 --sse-kms-key-id example-key-id --body filepath
```

Menggunakan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C)

Enkripsi sisi-server adalah tentang melindungi data diam. Enkripsi di sisi server hanya mengenkripsi data objek, bukan metadata objek. Dengan menggunakan enkripsi sisi server dengan kunci yang disediakan pelanggan (SSE-C), Anda dapat menyimpan data yang dienkripsi dengan kunci enkripsi Anda sendiri. Dengan kunci enkripsi yang Anda sediakan sebagai bagian dari permintaan Anda, Amazon S3 mengelola enkripsi data saat menulis ke disk dan dekripsi data saat Anda mengakses objek Anda. Oleh karena itu, Anda tidak perlu menyimpan kode apa pun untuk melakukan enkripsi dan dekripsi data. Satu-satunya hal yang perlu Anda lakukan adalah mengelola kunci enkripsi yang Anda berikan.

Saat Anda mengunggah objek, Amazon S3 menggunakan kunci enkripsi yang Anda berikan untuk menerapkan enkripsi AES-256 ke data Anda. Amazon S3 kemudian menghapus kunci enkripsi dari memori. Saat mengambil sebuah objek, Anda harus memberikan kunci enkripsi yang sama sebagai bagian dari permintaan Anda. Amazon S3 pertama-tama memverifikasi bahwa kunci enkripsi yang Anda berikan cocok, lalu mendekripsi objek sebelum mengembalikan data objek kepada Anda.

Tidak ada biaya tambahan untuk penggunaan SSE-C. Namun, permintaan untuk mengonfigurasi dan menggunakan SSE-C dikenakan biaya permintaan Amazon S3 standar. Untuk informasi tentang harga, lihat Harga Amazon S3.

> **Note**
>
> Amazon S3 tidak menyimpan kunci enkripsi yang Anda sediakan. Sebaliknya, ia menyimpan nilai Kode Autentikasi Pesan Berbasis Hash (HMAC) salted secara acak dari kunci enkripsi

untuk memvalidasi permintaan di masa mendatang. Nilai HMAC salted tidak dapat digunakan untuk mendapatkan nilai kunci enkripsi atau untuk mendekripsi konten objek terenkripsi. Artinya, jika Anda kehilangan kunci enkripsi, Anda akan kehilangan objek.

S3 Replication mendukung objek yang dienkripsi dengan SSE-C. Untuk informasi selengkapnya tentang mereplikasi objek terenkripsi, lihat the section called "Mereplikasi objek terenkripsi".

Untuk informasi selengkapnya tentang SSE-C, lihat topik berikut.

Topik

- Gambaran umum SSE-C
- Membutuhkan dan membatasi SSE-C
- URL dan SSE-C yang ditandatangani sebelumnya
- Menentukan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C)

Gambaran umum SSE-C

Bagian ini memberikan gambaran umum SSE-C. Saat menggunakan SSE-C, perhatikan pertimbangan berikut ini.

- Anda harus menggunakan HTTPS.

> **Important**
>
> Amazon S3 menolak permintaan apa pun yang dibuat melalui HTTP saat menggunakan SSE-C. Untuk pertimbangan keamanan, kami menyarankan Anda mempertimbangkan kunci apa pun yang salah Anda kirimkan melalui HTTP telah disusupi. Buang kuncinya dan putar seperlunya.

- Tag entitas (ETag) dalam respons bukanlah hash MD5 dari data objek.
- Anda mengelola pemetaan kunci enkripsi mana yang digunakan untuk mengenkripsi objek yang ingin dituju. Amazon S3 tidak menyimpan kunci enkripsi. Anda bertanggung jawab untuk melacak kunci enkripsi yang Anda berikan untuk objek yang ingin dituju.
  - Jika versi bucket Anda diaktifkan dengan Penentuan Versi, setiap versi objek yang Anda unggah menggunakan fitur ini dapat memiliki kunci enkripsinya sendiri. Anda bertanggung jawab untuk melacak kunci enkripsi yang Anda berikan untuk objek yang ingin dituju.

- Karena Anda mengelola kunci enkripsi di sisi klien, Anda mengelola perlindungan tambahan, seperti rotasi utama, di sisi klien.

> **Warning**
>
> Jika Anda kehilangan kunci enkripsi, setiap permintaan `GET` untuk objek tanpa kunci enkripsi akan gagal, dan Anda kehilangan objek tersebut.

Membutuhkan dan membatasi SSE-C

Untuk mewajibkan SSE-C bagi semua objek dalam bucket Amazon S3 tertentu, Anda dapat menggunakan kebijakan bucket.

Misalnya, kebijakan bucket berikut menolak izin unggah objek (`s3:PutObject`) untuk semua permintaan yang tidak menyertakan header `x-amz-server-side-encryption-customer-algorithm` yang meminta SSE-C.

```
{
    "Version": "2012-10-17",
    "Id": "PutObjectPolicy",
    "Statement": [
        {
            "Sid": "RequireSSECObjectUploads",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::example-s3-bucket/*",
            "Condition": {
                "Null": {
                    "s3:x-amz-server-side-encryption-customer-algorithm": "true"
                }
            }
        }
    ]
}
```

Anda juga dapat menggunakan kebijakan untuk membatasi enkripsi di sisi server pada semua objek di bucket Amazon S3 tertentu. Misalnya, kebijakan bucket berikut menolak izin unggah objek

(`s3:PutObject`) untuk semua permintaan yang tidak menyertakan header `x-amz-server-side-encryption-customer-algorithm` yang meminta SSE-C.

```
{
    "Version": "2012-10-17",
    "Id": "PutObjectPolicy",
    "Statement": [
        {
            "Sid": "RestrictSSECObjectUploads",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::example-s3-bucket/*",
            "Condition": {
                "Null": {
                    "s3:x-amz-server-side-encryption-customer-algorithm": "false"
                }
            }
        }
    ]
}
```

> **Important**
>
> Jika Anda menggunakan kebijakan bucket untuk mengharuskan SSE-C aktif`s3:PutObject`, Anda harus menyertakan `x-amz-server-side-encryption-customer-algorithm` header di semua permintaan unggahan multibagian (CreateMultipartUpload, UploadPart, dan). CompleteMultipartUpload

URL dan SSE-C yang ditandatangani sebelumnya

Anda dapat membuat URL yang telah ditandatangani sebelumnya yang dapat digunakan untuk operasi seperti mengunggah objek baru, mengambil objek yang sudah ada, atau mengambil metadata objek. URL yang ditandatangani sebelumnya mendukung SSE-C sebagai berikut:

- Saat membuat URL yang telah ditandatangani, Anda harus menentukan algoritma dengan menggunakan header `x-amz-server-side-encryption-customer-algorithm` dalam perhitungan tanda tangan.

- Saat menggunakan URL yang ditandatangani sebelumnya untuk mengunggah objek baru, mengambil objek yang sudah ada, atau mengambil metadata objek saja, Anda harus memberikan semua header enkripsi dalam permintaan aplikasi klien.

> **Note**
>
> Untuk objek non-SSE-C, Anda dapat membuat URL yang telah ditandatangani sebelumnya dan secara langsung merekatkan URL tersebut ke browser untuk mengakses data. Namun, Anda tidak dapat melakukan ini untuk objek SSE-C, karena selain URL yang ditandatangani sebelumnya, Anda juga harus menyertakan header HTTP yang kustom untuk objek SSE-C. Oleh karena itu, Anda hanya dapat menggunakan URL yang telah ditentukan sebelumnya untuk objek SSE-C secara terprogram.

Untuk informasi selengkapnya tentang URL yang telah ditandatangani, lihat the section called "Bekerja dengan URL yang telah ditandatangani".

Menentukan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C)

Pada saat pembuatan objek dengan API REST, Anda dapat menentukan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C). Saat Anda menggunakan SSE-C, Anda harus memberikan informasi kunci enkripsi menggunakan header permintaan berikut.

Nama	Deskripsi
`x-amz-server-side-encryption-customer-algorithm`	Gunakan header ini untuk menentukan algoritma enkripsi. Nilai header harus berupa `AES256`.
`x-amz-server-side-encryption-customer-key`	Gunakan header ini untuk menyediakan kunci enkripsi 256 bit, berkode base64 agar Amazon S3 dapat digunakan untuk mengenkripsi atau mendekripsi data Anda.
`x-amz-server-side-encryption-customer-key-MD5`	Gunakan header ini untuk menyediakan intisari kunci enkripsi MD5 128-bit yang dikodekan base64 menurut RFC 1321. Amazon S3 menggunakan header ini untuk pemeriksaan integritas pesan guna memastikan bahwa kunci enkripsi dikirimkan tanpa kesalahan.

Anda dapat menggunakan pustaka pembungkus AWS SDK untuk menambahkan header ini ke permintaan Anda. Jika perlu, Anda dapat melakukan panggilan API REST Amazon S3 secara langsung di aplikasi Anda.

> **Note**
>
> Anda tidak dapat menggunakan konsol Amazon S3 untuk mengunggah objek dan meminta SSE-C. Anda juga tidak dapat menggunakan konsol untuk memperbarui (misalnya, mengubah kelas penyimpanan atau menambahkan metadata) objek yang sudah ada yang disimpan menggunakan SSE-C.

Penggunaan API REST

Amazon S3 mendiamkan API rest yang mendukung SSE-C

API Amazon S3 berikut mendukung enkripsi di sisi server dengan kunci enkripsi yang disediakan pelanggan (SSE-C).

- GET operation–Saat mengambil objek menggunakan API GET (lihat GET Object), Anda dapat menentukan header permintaan.
- HEAD operation–Untuk mengambil metadata objek menggunakan API HEAD (lihat HEAD Object), Anda dapat menentukan header permintaan ini.
- PUT operation–Saat Anda mengunggah data dengan menggunakan API PUT Objek (lihat PUT Objek), Anda dapat menentukan header permintaan ini.
- Unggahan Multibagian–Saat mengunggah objek yang berukuran besar menggunakan API unggahan multibagian, Anda dapat menentukan header ini. Anda menentukan header ini dalam permintaan inisiasi (lihat Mulai Pengunggahan Multibagian) dan setiap permintaan pengunggahan bagian berikutnya (lihat Unggah Bagian atau Unggah Bagian-Salin). Untuk setiap permintaan unggahan bagian, informasi enkripsi harus sama dengan yang Anda berikan dalam permintaan unggahan multibagian.
- POST operation–Saat menggunakan POST operation untuk mengunggah objek (lihat Objek POST), alih-alih dengan header permintaan, Anda memberikan informasi yang sama di dalam bidang formulir.
- Copy operation–Saat Anda menyalin objek (lihat PUT Objek-Salin), Anda memiliki objek sumber dan objek target:

- Jika Anda ingin objek target dienkripsi menggunakan enkripsi sisi server dengan kunci AWS terkelola, Anda harus memberikan header permintaan. `x-amz-server-side-encryption`
- Jika Anda ingin objek target dienkripsi menggunakan SSE-C, Anda harus memberikan informasi enkripsi menggunakan tiga header yang dijelaskan dalam tabel sebelumnya.
- Jika objek sumber dienkripsi menggunakan SSE-C, Anda harus memberikan informasi kunci enkripsi menggunakan header berikut sehingga Amazon S3 dapat mendekripsi objek untuk disalin.

Nama	Deskripsi
`x-amz-copy-source-server-side-encryption-customer-algorithm`	Sertakan header ini untuk menentukan algoritma yang harus digunakan Amazon S3 untuk mendekripsi objek sumber. Nilai ini harus berupa `AES256`.
`x-amz-copy-source-server-side-encryption-customer-key`	Sertakan header ini untuk menyediakan kunci enkripsi berenkode base64 untuk Amazon S3 untuk digunakan untuk mendekripsi objek sumber. Kunci enkripsi ini harus berupa kunci yang Anda berikan kepada Amazon S3 saat membuat objek sumber. Jika tidak, Amazon S3 tidak dapat mendekripsi objeknya.
`x-amz-copy-source-server-side-encryption-customer-key-MD5`	Sertakan header ini untuk menyediakan intisari kunci enkripsi MD5 128-bit yang dikodekan base64 menurut RFC 1321.

Menggunakan AWS SDK untuk menentukan SSE-C untuk operasi PUT, GET, Head, dan Copy

Contoh berikut ini menunjukkan cara untuk meminta enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C) untuk objek. Contoh melakukan operasi berikut. Setiap operasi menunjukkan cara untuk menetapkan header terkait SSE-C pada permintaan:

- Put objek–Mengunggah objek dan meminta enkripsi di sisi server menggunakan kunci enkripsi yang disediakan pelanggan.

- Get object–Mengunduh objek yang diunggah dalam langkah sebelumnya. Dalam permintaan tersebut, Anda memberikan informasi enkripsi yang sama dengan yang Anda berikan saat unggah objek tersebut. Amazon S3 memerlukan informasi ini untuk mendekripsi objek sehingga dapat mengembalikannya kepada Anda.
- Get object metadata–Mengambil metadata objek. Anda memberikan informasi enkripsi yang sama, yang digunakan saat objek tersebut dibuat.
- Copy object–Membuat salinan dari objek yang diunggah sebelumnya. Karena objek sumber disimpan menggunakan SSE-C, Anda harus memberikan informasi enkripsinya dalam permintaan salinan Anda. Secara default, Amazon S3 mengenkripsi salinan objek hanya jika Anda secara eksplisit memintanya. Contoh ini mengarahkan Amazon S3 untuk menyimpan salinan objek yang dienkripsi.

Java

> **Note**
>
> Contoh ini menunjukkan cara untuk mengunggah objek dalam satu operasi. Saat menggunakan API Unggahan Multibagian untuk mengunggah objek besar, Anda memberikan informasi enkripsi dengan cara yang sama seperti yang ditunjukkan dalam contoh ini. Untuk contoh unggahan multipart yang menggunakan file AWS SDK for Java, lihat. Pengunggahan objek menggunakan unggahan multibagian

Untuk menambahkan informasi enkripsi yang diperlukan, Anda menyertakan `SSECustomerKey` dalam permintaan Anda. Untuk informasi selengkapnya tentang kelas `SSECustomerKey`, lihat bagian API REST.

Untuk informasi tentang SSE-C, lihat Menggunakan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C). Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

Example

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
```

```
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import javax.crypto.KeyGenerator;
import java.io.BufferedReader;
import java.io.File;
import java.io.IOException;
import java.io.InputStreamReader;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;

public class ServerSideEncryptionUsingClientSideEncryptionKey {
    private static SSECustomerKey SSE_KEY;
    private static AmazonS3 S3_CLIENT;
    private static KeyGenerator KEY_GENERATOR;

    public static void main(String[] args) throws IOException,
 NoSuchAlgorithmException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ***";
        String uploadFileName = "*** File path ***";
        String targetKeyName = "*** Target key name ***";

        // Create an encryption key.
        KEY_GENERATOR = KeyGenerator.getInstance("AES");
        KEY_GENERATOR.init(256, new SecureRandom());
        SSE_KEY = new SSECustomerKey(KEY_GENERATOR.generateKey());

        try {
            S3_CLIENT = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Upload an object.
            uploadObject(bucketName, keyName, new File(uploadFileName));

            // Download the object.
            downloadObject(bucketName, keyName);

            // Verify that the object is properly encrypted by attempting to
 retrieve it
```

```
            // using the encryption key.
            retrieveObjectMetadata(bucketName, keyName);

            // Copy the object into a new object that also uses SSE-C.
            copyObject(bucketName, keyName, targetKeyName);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    private static void uploadObject(String bucketName, String keyName, File file) {
        PutObjectRequest putRequest = new PutObjectRequest(bucketName, keyName,
file).withSSECustomerKey(SSE_KEY);
        S3_CLIENT.putObject(putRequest);
        System.out.println("Object uploaded");
    }

    private static void downloadObject(String bucketName, String keyName) throws
IOException {
        GetObjectRequest getObjectRequest = new GetObjectRequest(bucketName,
keyName).withSSECustomerKey(SSE_KEY);
        S3Object object = S3_CLIENT.getObject(getObjectRequest);

        System.out.println("Object content: ");
        displayTextInputStream(object.getObjectContent());
    }

    private static void retrieveObjectMetadata(String bucketName, String keyName) {
        GetObjectMetadataRequest getMetadataRequest = new
GetObjectMetadataRequest(bucketName, keyName)
                .withSSECustomerKey(SSE_KEY);
        ObjectMetadata objectMetadata =
S3_CLIENT.getObjectMetadata(getMetadataRequest);
        System.out.println("Metadata retrieved. Object size: " +
objectMetadata.getContentLength());
    }
```

```
    private static void copyObject(String bucketName, String keyName, String
 targetKeyName)
            throws NoSuchAlgorithmException {
        // Create a new encryption key for target so that the target is saved using
        // SSE-C.
        SSECustomerKey newSSEKey = new SSECustomerKey(KEY_GENERATOR.generateKey());

        CopyObjectRequest copyRequest = new CopyObjectRequest(bucketName, keyName,
 bucketName, targetKeyName)
                .withSourceSSECustomerKey(SSE_KEY)
                .withDestinationSSECustomerKey(newSSEKey);

        S3_CLIENT.copyObject(copyRequest);
        System.out.println("Object copied");
    }

    private static void displayTextInputStream(S3ObjectInputStream input) throws
 IOException {
        // Read one line at a time from the input stream and display each line.
        BufferedReader reader = new BufferedReader(new InputStreamReader(input));
        String line;
        while ((line = reader.readLine()) != null) {
            System.out.println(line);
        }
        System.out.println();
    }
}
```

.NET

> **Note**
>
> Untuk contoh pengunggahan objek besar menggunakan API unggahan multibagian, lihat Pengunggahan objek menggunakan unggahan multibagian dan Menggunakan AWS SDK (API tingkat rendah).

Untuk informasi tentang SSE-C, lihat Menggunakan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C). Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

Example

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.IO;
using System.Security.Cryptography;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class SSEClientEncryptionKeyObjectOperationsTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** key name for new object created ***";
        private const string copyTargetKeyName = "*** key name for object copy ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            ObjectOpsUsingClientEncryptionKeyAsync().Wait();
        }
        private static async Task ObjectOpsUsingClientEncryptionKeyAsync()
        {
            try
            {
                // Create an encryption key.
                Aes aesEncryption = Aes.Create();
                aesEncryption.KeySize = 256;
                aesEncryption.GenerateKey();
                string base64Key = Convert.ToBase64String(aesEncryption.Key);

                // 1. Upload the object.
                PutObjectRequest putObjectRequest = await
 UploadObjectAsync(base64Key);
                // 2. Download the object and verify that its contents matches what
 you uploaded.
                await DownloadObjectAsync(base64Key, putObjectRequest);
```

```
                // 3. Get object metadata and verify that the object uses AES-256
encryption.
                await GetObjectMetadataAsync(base64Key);
                // 4. Copy both the source and target objects using server-side
encryption with
                //    a customer-provided encryption key.
                await CopyObjectAsync(aesEncryption, base64Key);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered ***. Message:'{0}' when writing
an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
        }

        private static async Task<PutObjectRequest> UploadObjectAsync(string
base64Key)
        {
            PutObjectRequest putObjectRequest = new PutObjectRequest
            {
                BucketName = bucketName,
                Key = keyName,
                ContentBody = "sample text",
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key
            };
            PutObjectResponse putObjectResponse = await
client.PutObjectAsync(putObjectRequest);
            return putObjectRequest;
        }
        private static async Task DownloadObjectAsync(string base64Key,
PutObjectRequest putObjectRequest)
        {
            GetObjectRequest getObjectRequest = new GetObjectRequest
            {
                BucketName = bucketName,
                Key = keyName,
```

```
                // Provide encryption information for the object stored in Amazon
S3.
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key
            };

            using (GetObjectResponse getResponse = await
client.GetObjectAsync(getObjectRequest))
            using (StreamReader reader = new
StreamReader(getResponse.ResponseStream))
            {
                string content = reader.ReadToEnd();
                if (String.Compare(putObjectRequest.ContentBody, content) == 0)
                    Console.WriteLine("Object content is same as we uploaded");
                else
                    Console.WriteLine("Error...Object content is not same.");

                if (getResponse.ServerSideEncryptionCustomerMethod ==
ServerSideEncryptionCustomerMethod.AES256)
                    Console.WriteLine("Object encryption method is AES256, same as
we set");
                else
                    Console.WriteLine("Error...Object encryption method is not the
same as AES256 we set");

                // Assert.AreEqual(putObjectRequest.ContentBody, content);
                // Assert.AreEqual(ServerSideEncryptionCustomerMethod.AES256,
getResponse.ServerSideEncryptionCustomerMethod);
            }
        }
        private static async Task GetObjectMetadataAsync(string base64Key)
        {
            GetObjectMetadataRequest getObjectMetadataRequest = new
GetObjectMetadataRequest
            {
                BucketName = bucketName,
                Key = keyName,

                // The object stored in Amazon S3 is encrypted, so provide the
necessary encryption information.
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key
```

```
            };

            GetObjectMetadataResponse getObjectMetadataResponse = await
 client.GetObjectMetadataAsync(getObjectMetadataRequest);
            Console.WriteLine("The object metadata show encryption method used is:
 {0}", getObjectMetadataResponse.ServerSideEncryptionCustomerMethod);
            // Assert.AreEqual(ServerSideEncryptionCustomerMethod.AES256,
 getObjectMetadataResponse.ServerSideEncryptionCustomerMethod);
        }
        private static async Task CopyObjectAsync(Aes aesEncryption, string
 base64Key)
        {
            aesEncryption.GenerateKey();
            string copyBase64Key = Convert.ToBase64String(aesEncryption.Key);

            CopyObjectRequest copyRequest = new CopyObjectRequest
            {
                SourceBucket = bucketName,
                SourceKey = keyName,
                DestinationBucket = bucketName,
                DestinationKey = copyTargetKeyName,
                // Information about the source object's encryption.
                CopySourceServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
                CopySourceServerSideEncryptionCustomerProvidedKey = base64Key,
                // Information about the target object's encryption.
                ServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = copyBase64Key
            };
            await client.CopyObjectAsync(copyRequest);
        }
    }
}
```

Menggunakan AWS SDK untuk menentukan SSE-C untuk unggahan multipart

Contoh di bagian sebelumnya menunjukkan cara meminta enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C) dalam operasi PUT, GET, Head, dan Copy. Bagian ini menjelaskan API Amazon S3 lainnya yang mendukung SSE-C.

Java

Untuk mengunggah objek besar, Anda dapat menggunakan API unggahan multibagian (lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian). Anda dapat menggunakan API tingkat tinggi atau tingkat rendah untuk mengunggah objek besar. API ini mendukung header terkait enkripsi dalam permintaan.

- Saat menggunakan API `TransferManager` tingkat tinggi, Anda menyediakan header kustom enkripsi di `PutObjectRequest` (lihat Pengunggahan objek menggunakan unggahan multibagian).
- Saat menggunakan API tingkat rendah, Anda memberikan informasi terkait enkripsi pada `InitiateMultipartUploadRequest`, diikuti dengan informasi enkripsi identik pada setiap `UploadPartRequest`. Anda tidak perlu memberikan header kustom enkripsi apa pun dalam `CompleteMultipartUploadRequest` Anda. Sebagai contoh, lihat Menggunakan AWS SDK (API tingkat rendah).

Contoh berikut menggunakan `TransferManager` untuk membuat objek dan menunjukkan cara memberikan informasi terkait SSE-C. Contoh ini melakukan hal berikut:

- Membuat objek menggunakan metode `TransferManager.upload()`. Di dalam instans `PutObjectRequest`, Anda memberikan informasi kunci enkripsi untuk meminta. Amazon S3 mengenkripsi objek menggunakan kunci yang disediakan pelanggan.
- Membuat salinan objek dengan memanggil metode `TransferManager.copy()`. Contoh tersebut mengarahkan Amazon S3 untuk mengenkripsi salinan objek menggunakan `SSECustomerKey` yang baru. Karena objek sumber dienkripsi menggunakan SSE-C, `CopyObjectRequest` juga menyediakan kunci enkripsi objek sumber sehingga Amazon S3 dapat mendekripsi objek tersebut sebelum menyalinnya.

Example

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CopyObjectRequest;
```

```
import com.amazonaws.services.s3.model.PutObjectRequest;
import com.amazonaws.services.s3.model.SSECustomerKey;
import com.amazonaws.services.s3.transfer.Copy;
import com.amazonaws.services.s3.transfer.TransferManager;
import com.amazonaws.services.s3.transfer.TransferManagerBuilder;
import com.amazonaws.services.s3.transfer.Upload;

import javax.crypto.KeyGenerator;
import java.io.File;
import java.security.SecureRandom;

public class ServerSideEncryptionCopyObjectUsingHLwithSSEC {

    public static void main(String[] args) throws Exception {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String fileToUpload = "*** File path ***";
        String keyName = "*** New object key name ***";
        String targetKeyName = "*** Key name for object copy ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();
            TransferManager tm = TransferManagerBuilder.standard()
                    .withS3Client(s3Client)
                    .build();

            // Create an object from a file.
            PutObjectRequest putObjectRequest = new PutObjectRequest(bucketName,
 keyName, new File(fileToUpload));

            // Create an encryption key.
            KeyGenerator keyGenerator = KeyGenerator.getInstance("AES");
            keyGenerator.init(256, new SecureRandom());
            SSECustomerKey sseCustomerEncryptionKey = new
 SSECustomerKey(keyGenerator.generateKey());

            // Upload the object. TransferManager uploads asynchronously, so this
 call
            // returns immediately.
            putObjectRequest.setSSECustomerKey(sseCustomerEncryptionKey);
            Upload upload = tm.upload(putObjectRequest);
```

```
            // Optionally, wait for the upload to finish before continuing.
            upload.waitForCompletion();
            System.out.println("Object created.");

            // Copy the object and store the copy using SSE-C with a new key.
            CopyObjectRequest copyObjectRequest = new CopyObjectRequest(bucketName,
 keyName, bucketName, targetKeyName);
            SSECustomerKey sseTargetObjectEncryptionKey = new
 SSECustomerKey(keyGenerator.generateKey());
            copyObjectRequest.setSourceSSECustomerKey(sseCustomerEncryptionKey);

 copyObjectRequest.setDestinationSSECustomerKey(sseTargetObjectEncryptionKey);

            // Copy the object. TransferManager copies asynchronously, so this call
 returns
            // immediately.
            Copy copy = tm.copy(copyObjectRequest);

            // Optionally, wait for the upload to finish before continuing.
            copy.waitForCompletion();
            System.out.println("Copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Untuk mengunggah objek besar, Anda dapat menggunakan API unggahan multibagian (lihatMengunggah dan menyalin objek menggunakan unggahan multibagian). AWS SDK for .NET menyediakan API tingkat tinggi atau tingkat rendah untuk mengunggah objek besar. API ini mendukung header terkait enkripsi dalam permintaan.

- Saat menggunakan API `Transfer-Utility` tingkat tinggi, Anda menyediakan header kustom enkripsi di `TransferUtilityUploadRequest` seperti yang ditunjukkan. Untuk contoh kode, lihat Pengunggahan objek menggunakan unggahan multibagian.

```
TransferUtilityUploadRequest request = new TransferUtilityUploadRequest()
{
    FilePath = filePath,
    BucketName = existingBucketName,
    Key = keyName,
    // Provide encryption information.
    ServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
    ServerSideEncryptionCustomerProvidedKey = base64Key,
};
```

- Saat menggunakan API tingkat rendah, Anda memberikan informasi terkait enkripsi dalam memulai permintaan unggahan multibagian, diikuti dengan informasi enkripsi yang sama dalam permintaan bagian unggahan berikutnya. Anda tidak perlu memberikan header kustom enkripsi apa pun dalam permintaan pengunggahan multibagian lengkap Anda. Sebagai contoh, lihat Menggunakan AWS SDK (API tingkat rendah).

  Berikut ini adalah contoh pengunggahan multibagian tingkat rendah yang membuat salinan objek besar yang sudah ada. Pada contoh, objek yang akan disalin disimpan di Amazon S3 menggunakan SSE-C, dan Anda ingin menyimpan objek target juga menggunakan SSE-C. Dalam contoh ini, Anda melakukan hal berikut:

  - Mulai permintaan pengunggahan multibagian dengan memberikan kunci enkripsi dan informasi terkait.
  - Menyediakan kunci enkripsi objek sumber dan sasaran serta informasi terkait dalam `CopyPartRequest`.
  - Dapatkan ukuran objek sumber yang akan disalin dengan mengambil metadata objek.
  - Unggah objek ke dalam bagian 5 MB.

  Example

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
```

```
using System.IO;
using System.Security.Cryptography;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class SSECLowLevelMPUcopyObjectTest
    {
        private const string existingBucketName = "*** bucket name ***";
        private const string sourceKeyName      = "*** source object key name
 ***";
        private const string targetKeyName      = "*** key name for the target
 object ***";
        private const string filePath           = @"*** file path ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            CopyObjClientEncryptionKeyAsync().Wait();
        }

        private static async Task CopyObjClientEncryptionKeyAsync()
        {
            Aes aesEncryption = Aes.Create();
            aesEncryption.KeySize = 256;
            aesEncryption.GenerateKey();
            string base64Key = Convert.ToBase64String(aesEncryption.Key);

            await CreateSampleObjUsingClientEncryptionKeyAsync(base64Key,
 s3Client);

            await CopyObjectAsync(s3Client, base64Key);
        }
        private static async Task CopyObjectAsync(IAmazonS3 s3Client, string
 base64Key)
        {
            List<CopyPartResponse> uploadResponses = new List<CopyPartResponse>();

            // 1. Initialize.
            InitiateMultipartUploadRequest initiateRequest = new
 InitiateMultipartUploadRequest
```

```
            {
                BucketName = existingBucketName,
                Key = targetKeyName,
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key,
            };

            InitiateMultipartUploadResponse initResponse =
                await s3Client.InitiateMultipartUploadAsync(initiateRequest);

            // 2. Upload Parts.
            long partSize = 5 * (long)Math.Pow(2, 20); // 5 MB
            long firstByte = 0;
            long lastByte = partSize;

            try
            {
                // First find source object size. Because object is stored
encrypted with
                // customer provided key you need to provide encryption
information in your request.
                GetObjectMetadataRequest getObjectMetadataRequest = new
GetObjectMetadataRequest()
                {
                    BucketName = existingBucketName,
                    Key = sourceKeyName,
                    ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                    ServerSideEncryptionCustomerProvidedKey = base64Key // " *
**source object encryption key ***"
                };

                GetObjectMetadataResponse getObjectMetadataResponse = await
s3Client.GetObjectMetadataAsync(getObjectMetadataRequest);

                long filePosition = 0;
                for (int i = 1; filePosition <
getObjectMetadataResponse.ContentLength; i++)
                {
                    CopyPartRequest copyPartRequest = new CopyPartRequest
                    {
                        UploadId = initResponse.UploadId,
                        // Source.
```

```
                        SourceBucket = existingBucketName,
                        SourceKey = sourceKeyName,
                        // Source object is stored using SSE-C. Provide encryption
information.
                        CopySourceServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                        CopySourceServerSideEncryptionCustomerProvidedKey =
base64Key, //"***source object encryption key ***",
                        FirstByte = firstByte,
                        // If the last part is smaller then our normal part size
then use the remaining size.
                        LastByte = lastByte >
getObjectMetadataResponse.ContentLength ?
                            getObjectMetadataResponse.ContentLength - 1 :
lastByte,

                        // Target.
                        DestinationBucket = existingBucketName,
                        DestinationKey = targetKeyName,
                        PartNumber = i,
                        // Encryption information for the target object.
                        ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                        ServerSideEncryptionCustomerProvidedKey = base64Key
                    };
                    uploadResponses.Add(await
s3Client.CopyPartAsync(copyPartRequest));
                    filePosition += partSize;
                    firstByte += partSize;
                    lastByte += partSize;
                }

                // Step 3: complete.
                CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest
                {
                    BucketName = existingBucketName,
                    Key = targetKeyName,
                    UploadId = initResponse.UploadId,
                };
                completeRequest.AddPartETags(uploadResponses);

                CompleteMultipartUploadResponse completeUploadResponse =
                    await s3Client.CompleteMultipartUploadAsync(completeRequest);
```

```
            }
            catch (Exception exception)
            {
                Console.WriteLine("Exception occurred: {0}", exception.Message);
                AbortMultipartUploadRequest abortMPURequest = new
AbortMultipartUploadRequest
                {
                    BucketName = existingBucketName,
                    Key = targetKeyName,
                    UploadId = initResponse.UploadId
                };
                s3Client.AbortMultipartUpload(abortMPURequest);
            }
        }
        private static async Task
CreateSampleObjUsingClientEncryptionKeyAsync(string base64Key, IAmazonS3
s3Client)
        {
            // List to store upload part responses.
            List<UploadPartResponse> uploadResponses = new
List<UploadPartResponse>();

            // 1. Initialize.
            InitiateMultipartUploadRequest initiateRequest = new
InitiateMultipartUploadRequest
            {
                BucketName = existingBucketName,
                Key = sourceKeyName,
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key
            };

            InitiateMultipartUploadResponse initResponse =
               await s3Client.InitiateMultipartUploadAsync(initiateRequest);

            // 2. Upload Parts.
            long contentLength = new FileInfo(filePath).Length;
            long partSize = 5 * (long)Math.Pow(2, 20); // 5 MB

            try
            {
                long filePosition = 0;
                for (int i = 1; filePosition < contentLength; i++)
```

```
                {
                    UploadPartRequest uploadRequest = new UploadPartRequest
                    {
                        BucketName = existingBucketName,
                        Key = sourceKeyName,
                        UploadId = initResponse.UploadId,
                        PartNumber = i,
                        PartSize = partSize,
                        FilePosition = filePosition,
                        FilePath = filePath,
                        ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                        ServerSideEncryptionCustomerProvidedKey = base64Key
                    };

                    // Upload part and add response to our list.
                    uploadResponses.Add(await
s3Client.UploadPartAsync(uploadRequest));

                    filePosition += partSize;
                }

                // Step 3: complete.
                CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest
                {
                    BucketName = existingBucketName,
                    Key = sourceKeyName,
                    UploadId = initResponse.UploadId,
                    //PartETags = new List<PartETag>(uploadResponses)

                };
                completeRequest.AddPartETags(uploadResponses);

                CompleteMultipartUploadResponse completeUploadResponse =
                    await s3Client.CompleteMultipartUploadAsync(completeRequest);

            }
            catch (Exception exception)
            {
                Console.WriteLine("Exception occurred: {0}", exception.Message);
                AbortMultipartUploadRequest abortMPURequest = new
AbortMultipartUploadRequest
                {
```

```
                    BucketName = existingBucketName,
                    Key = sourceKeyName,
                    UploadId = initResponse.UploadId
                };
                await s3Client.AbortMultipartUploadAsync(abortMPURequest);
            }
        }
    }
}
```

Melindungi data menggunakan enkripsi di sisi klien

Enkripsi di sisi klien adalah tindakan mengenkripsi data Anda secara lokal untuk membantu memastikan keamanannya saat sedang transit dan saat disimpan. Untuk mengenkripsi objek Anda sebelum mengirimnya ke Amazon S3, gunakan Klien Enkripsi Amazon S3. Ketika objek Anda dienkripsi dengan cara ini, objek Anda tidak terpapar ke pihak ketiga mana pun, termasuk. AWS Amazon S3 menerima objek Anda yang sudah dienkripsi; Amazon S3 tidak berperan dalam mengenkripsi atau mendekripasi objek Anda. Anda dapat menggunakan Klien Enkripsi Amazon S3 dan enkripsi di sisi server untuk mengenkripsi data Anda. Saat Anda mengirim objek terenkripsi ke Amazon S3, Amazon S3 tidak mengenali objek sebagai dienkripsi, hanya mendeteksi objek biasa.

Klien Enkripsi Amazon S3 berfungsi sebagai perantara antara Anda dan Amazon S3. Setelah Anda membuat instans Klien Enkripsi Amazon S3, objek Anda secara otomatis dienkripsi dan didekripsi sebagai bagian dari Amazon S3 dan permintaan `PutObject` dan `GetObject` Anda. Semua objek Anda dienkripsi dengan kunci data unik. Klien Enkripsi Amazon S3 tidak menggunakan atau berinteraksi dengan kunci bucket, meskipun Anda menentukan kunci KMS sebagai kunci pembungkus.

Panduan Pengembang Klien Enkripsi Amazon S3 berfokus pada versi 3.0 dan yang lebih baru dari Klien Enkripsi Amazon S3. Untuk informasi selengkapnya, lihat Apa itu Klien Enkripsi Amazon S3? dalam Panduan Pengembang Klien Enkripsi Amazon S3.

Untuk informasi selengkapnya tentang versi klien Enkripsi Amazon S3 sebelumnya, lihat Panduan Pengembang AWS SDK untuk bahasa pemrograman Anda.

- AWS SDK for Java
- AWS SDK for .NET
- AWS SDK for Go

- AWS SDK for PHP
- AWS SDK for Ruby
- AWS SDK for C++

Privasi lalu lintas antar jaringan

Topik ini menjelaskan cara Amazon S3 mengamankan koneksi dari layanan ke lokasi lain.

Lalu lintas antara layanan dan aplikasi serta klien on-premise

Koneksi berikut dapat digabungkan dengan AWS PrivateLink untuk menyediakan konektivitas antara jaringan pribadi Anda dan AWS:

- Koneksi AWS VPN Site-to-Site. Untuk informasi lebih lanjut, lihat Apa itu AWS Site-to-Site VPN?
- AWS Direct Connect Koneksi. Untuk informasi lebih lanjut, lihat Apa itu AWS Direct Connect?

Akses ke Amazon S3 melalui jaringan adalah melalui API yang AWS dipublikasikan. Klien harus mendukung Keamanan Lapisan Pengangkutan (TLS) 1.2. Kami merekomendasikan TLS 1.3. Klien juga harus mendukung suite cipher dengan perfect forward secrecy (PFS) seperti Ephemeral Diffie-Hellman (DHE) atau Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Sebagian besar sistem modern seperti Java 7 dan versi yang lebih baru support mode ini. Selain itu, Anda harus menandatangani permintaan menggunakan kunci akses ID dan kunci akses rahasia yang terkait dengan IAM pengguna utama, atau Anda dapat menggunakan AWS Security Token Service (STS) untuk membuat kredensial keamanan sementara guna menandatangani permintaan.

Lalu lintas antar AWS sumber daya di Wilayah yang sama

Titik akhir cloud privat virtual (VPC) untuk Amazon S3 adalah entitas logis dalam sebuah VPC yang mengizinkan konektivitas hanya ke Amazon S3. Amazon VPC merutekan permintaan ke Amazon S3 dan merutekan respons kembali ke VPC. Untuk informasi selengkapnya, lihat Titik Akhir VPC di Panduan Pengguna VPC. Untuk contoh kebijakan bucket yang dapat Anda gunakan untuk mengontrol akses bucket S3 dari titik akhir VPC, lihat Megendalikan akses dari titik akhir VPC dengan kebijakan bucket.

AWS PrivateLink untuk Amazon S3

Dengan AWS PrivateLink Amazon S3, Anda dapat menyediakan titik akhir VPC antarmuka (titik akhir antarmuka) di cloud pribadi virtual (VPC) Anda. Titik akhir ini dapat diakses secara langsung dari aplikasi yang ada di tempat melalui VPN dan AWS Direct Connect, atau dalam pengintipan VPC yang berbeda Wilayah AWS .

Titik akhir antarmuka diwakili oleh satu atau lebih antarmuka jaringan elastis (ENI) yang ditugaskan alamat IP privat dari subnet di VPC Anda. Permintaan ke Amazon S3 melalui titik akhir antarmuka tetap berada di jaringan Amazon. Anda juga dapat mengakses titik akhir antarmuka di VPC Anda dari aplikasi lokal AWS Direct Connect melalui AWS Virtual Private Network atau ().AWS VPN Untuk informasi selengkapnya tentang cara menghubungkan VPC dengan jaringan on-premise Anda, lihat AWS Direct Connect Panduan Pengguna dan AWS Site-to-Site VPN Panduan Pengguna.

Untuk informasi umum tentang titik akhir antarmuka, lihat Antarmuka titik akhir VPC (AWS PrivateLink) dalam Panduan AWS PrivateLink .

Topik

- Jenis titik akhir VPC untuk Amazon S3
- Pembatasan dan batasan AWS PrivateLink untuk Amazon S3
- Membuat titik akhir VPC
- Mengakses titik akhir antarmuka Amazon S3
- DNS privat
- Mengakses bucket, titik akses, serta operasi API Amazon S3 Control dari titik akhir antarmuka S3
- Memperbarui konfigurasi DNS on-premise
- Membuat kebijakan titik akhir VPC untuk Amazon S3

Jenis titik akhir VPC untuk Amazon S3

Anda dapat menggunakan dua jenis titik akhir VPC untuk mengakses Amazon S3: titik akhir gateway dan titik akhir antarmuka (dengan menggunakan). AWS PrivateLinkTitik akhir gateway adalah gateway yang Anda tentukan dalam tabel rute untuk mengakses Amazon S3 dari VPC melalui jaringan. AWS Titik akhir antarmuka memperluas fungsionalitas titik akhir gateway dengan menggunakan alamat IP pribadi untuk merutekan permintaan ke Amazon S3 dari dalam VPC Anda, di tempat, atau dari VPC di tempat lain dengan menggunakan peering VPC atau. Wilayah AWS AWS

Transit Gateway Untuk informasi selengkapnya, lihat Apa itu peering VPC? dan Transit Gateway vs Peering VPC.

Titik akhir antarmuka kompatibel dengan titik akhir gateway. Jika Anda memiliki titik akhir gateway yang ada di VPC, Anda dapat menggunakan kedua jenis titik akhir di VPC yang sama.

Titik akhir gateway untuk Amazon S3	Titik akhir antarmuka untuk Amazon S3
Dalam kedua kasus, lalu lintas jaringan Anda tetap berada di AWS jaringan.	
Gunakan alamat IP publik Amazon S3	Gunakan alamat IP privat dari VPC Anda untuk mengakses Amazon S3
Gunakan nama Amazon S3 DNS yang sama	Memerlukan nama DNS Amazon S3 kustom titik akhir
Jangan izinkan mengakses dari on-premise	Izinkan mengakses dari on-premise
Jangan izinkan akses dari yang lain Wilayah AWS	Izinkan akses dari VPC di VPC lain Wilayah AWS dengan menggunakan VPC peering atau AWS Transit Gateway
Tidak ditagih	Ditagih

Untuk informasi selengkapnya tentang titik akhir gateway, lihat Titik akhir VPC Gateway dalam Panduan AWS PrivateLink .

Pembatasan dan batasan AWS PrivateLink untuk Amazon S3

Batasan VPC berlaku AWS PrivateLink untuk Amazon S3. Untuk informasi selengkapnya, lihat Pertimbangan antarmuka titik akhir dan AWS PrivateLink kuota dalam AWS PrivateLink Panduan. Selain itu, larangan berikut juga berlaku.

AWS PrivateLink untuk Amazon S3 tidak mendukung yang berikut:

- Titik Akhir Standar Proses Informasi Federal (FIPS)
- Titik akhir situs web
- Titik akhir warisan global
- Titik akhir Wilayah dash S3

- Titik akhir tumpukan ganda Amazon S3
- Menggunakan CopyObjectatau UploadPartCopydi antara ember dalam berbagai Wilayah AWS
- Keamanan Lapisan Pengangkutan (TLS) 1.1

Membuat titik akhir VPC

Untuk membuat titik akhir antarmuka VPC, lihat Membuat titik akhir VPC dalam AWS PrivateLink Panduan.

Mengakses titik akhir antarmuka Amazon S3

Saat Anda membuat titik akhir antarmuka, Amazon S3 menghasilkan dua jenis nama DNS S3 khusus titik akhir: Regional dan zonal.

- Nama DNS Regional mencakup ID titik akhir VPC unik, pengenal layanan, Wilayah AWS dan namanya. `vpce.amazonaws.com` Sebagai contoh, untuk ID titik akhir VPC *`vpce-1a2b3c4d`*, nama DNS yang dihasilkan mungkin mirip dengan *`vpce-1a2b3c4d-5e6f`*`.s3.us-east-1.vpce.amazonaws.com`.
- Nama DNS Zonal mencakup Zona Ketersediaan—misalnya, *`vpce-1a2b3c4d-5e6f`***`-us-east-1a.`**`s3.us-east-1.vpce.amazonaws.com`. Anda dapat menggunakan opsi ini jika arsitektur Anda mengisolasi Zona Ketersediaan. Contoh, Anda bisa menggunakannya untuk kontainer kesalahan atau untuk mengurangi biaya transfer data Regional.

Nama DNS S3 spesifik titik akhir dapat diselesaikan dari domain DNS publik S3.

DNS privat

Opsi DNS privat untuk titik akhir antarmuka VPC menyederhanakan perutean lalu lintas S3 melalui titik akhir VPC dan membantu Anda memanfaatkan jalur jaringan dengan biaya terendah yang tersedia untuk aplikasi Anda. Anda dapat menggunakan opsi DNS privat untuk merutekan lalu lintas S3 Regional tanpa memperbarui klien S3 Anda untuk menggunakan nama DNS spesifik titik akhir dari titik akhir antarmuka Anda, atau mengelola infrastruktur DNS. Dengan mengaktifkan nama DNS pribadi, kueri DNS S3 Regional menyelesaikan ke alamat IP pribadi AWS PrivateLink untuk titik akhir berikut:

- Titik akhir bucket regional (misalnya, `s3.us-east-1.amazonaws.com`)
- Kontrol titik akhir (misalnya, `s3-control.us-east-1.amazonaws.com`)

- Titik akhir titik akses (misalnya, `s3-accesspoint.us-east-1.amazonaws.com`)

Jika Anda memiliki titik akhir gateway di VPC Anda, Anda dapat secara otomatis merutekan permintaan dalam VPC melalui titik akhir gateway S3 yang ada dan on-premise melalui titik akhir antarmuka Anda. Pendekatan ini memungkinkan Anda untuk mengoptimalkan biaya jaringan Anda dengan menggunakan titik akhir gateway, yang tidak ditagih, untuk lalu lintas in-VPC Anda. Aplikasi lokal Anda dapat digunakan AWS PrivateLink dengan bantuan titik akhir Resolver masuk. Amazon menyediakan server DNS, yang disebut Resolver Route 53, untuk VPC Anda. Titik akhir Resolver masuk meneruskan kueri DNS dari jaringan on-premise ke Resolver Route 53.

> **Important**
>
> Untuk memanfaatkan jalur jaringan dengan biaya terendah saat menggunakan Aktifkan DNS pribadi hanya untuk titik akhir masuk, titik akhir gateway harus ada di VPC Anda. Kehadiran titik akhir gateway membantu memastikan bahwa lalu lintas dalam VPC selalu merutekan melalui jaringan AWS pribadi saat opsi Aktifkan DNS pribadi hanya untuk titik akhir masuk dipilih. Anda harus mempertahankan titik akhir gateway ini sementara Anda memiliki opsi Aktifkan DNS pribadi hanya untuk titik akhir masuk yang dipilih. Jika Anda ingin menghapus titik akhir gateway Anda, Anda harus terlebih dahulu menghapus Aktifkan DNS pribadi hanya untuk titik akhir masuk.
>
> Jika Anda ingin memperbarui titik akhir antarmuka yang ada ke Aktifkan DNS pribadi hanya untuk titik akhir masuk, konfirmasikan terlebih dahulu bahwa VPC Anda memiliki titik akhir gateway S3. Untuk informasi selengkapnya tentang titik akhir gateway dan mengelola nama DNS privat, lihat Titik akhir VPC Gateway dan Kelola nama DNS masing-masing di Panduan AWS PrivateLink .

Opsi Aktifkan DNS pribadi hanya untuk titik akhir masuk hanya tersedia untuk layanan yang mendukung titik akhir gateway.

Untuk informasi selengkapnya tentang membuat titik akhir VPC yang menggunakan Aktifkan DNS privat hanya untuk titik akhir yang masuk, lihat Membuat titik akhir antarmuka di Panduan AWS PrivateLink .

Menggunakan konsol VPC

Di konsol Anda memiliki dua opsi: Aktifkan nama DNS dan Aktifkan DNS pribadi hanya untuk titik akhir masuk. Aktifkan nama DNS adalah opsi yang didukung oleh AWS PrivateLink. Dengan

menggunakan opsi Aktifkan nama DNS, Anda dapat menggunakan konektivitas pribadi Amazon ke Amazon S3, sambil membuat permintaan ke nama DNS titik akhir publik default. Ketika opsi ini diaktifkan, pelanggan dapat memanfaatkan jalur jaringan biaya terendah yang tersedia untuk aplikasi mereka.

Saat Anda mengaktifkan nama DNS pribadi pada titik akhir antarmuka VPC yang ada atau baru untuk Amazon S3, opsi Aktifkan DNS pribadi hanya untuk titik akhir masuk dipilih secara default. Jika opsi ini dipilih, aplikasi Anda hanya menggunakan titik akhir antarmuka untuk lalu lintas on-premise Anda. Lalu lintas in-VPC ini secara otomatis menggunakan titik akhir gateway berbiaya lebih rendah. Atau, Anda dapat menghapus Aktifkan DNS pribadi hanya untuk titik akhir masuk untuk merutekan semua permintaan S3 melalui titik akhir antarmuka Anda.

Menggunakan AWS CLI

Jika Anda tidak menentukan nilai untuk `PrivateDnsOnlyForInboundResolverEndpoint`, itu akan secara default diatur ke `true`. Namun, sebelum VPC Anda menerapkan pengaturan Anda, VPC melakukan pemeriksaan untuk memastikan bahwa Anda memiliki titik akhir gateway yang ada di VPC. Jika titik akhir gateway ada di VPC, panggilan berhasil. Jika tidak, Anda akan melihat pesan kesalahan berikut ini:

Untuk disetel PrivateDnsOnlyForInboundResolverEndpoint ke true, VPC *`vpce_id`* harus memiliki titik akhir gateway untuk layanan.

Untuk titik akhir Antarmuka VPC baru

Gunakan atribut `private-dns-enabled` dan `dns-options` untuk mengaktifkan DNS pribadi melalui baris perintah. Opsi `PrivateDnsOnlyForInboundResolverEndpoint` dalam atribut `dns-options` harus diatur ke `true`. Ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
aws ec2 create-vpc-endpoint \
--region us-east-1 \
--service-name s3-service-name \
--vpc-id client-vpc-id \
--subnet-ids client-subnet-id \
--vpc-endpoint-type Interface  \
--private-dns-enabled  \
--ip-address-type ip-address-type \
--dns-options PrivateDnsOnlyForInboundResolverEndpoint=true \
```

```
--security-group-ids client-sg-id
```

Untuk titik akhir VPC yang ada

Jika Anda ingin menggunakan DNS pribadi untuk titik akhir VPC yang ada, gunakan perintah contoh berikut dan ganti dengan informasi Anda *user input placeholders* sendiri.

```
aws ec2 modify-vpc-endpoint \
--region us-east-1 \
--vpc-endpoint-id client-vpc-id \
--private-dns-enabled \
--dns-options PrivateDnsOnlyForInboundResolverEndpoint=false
```

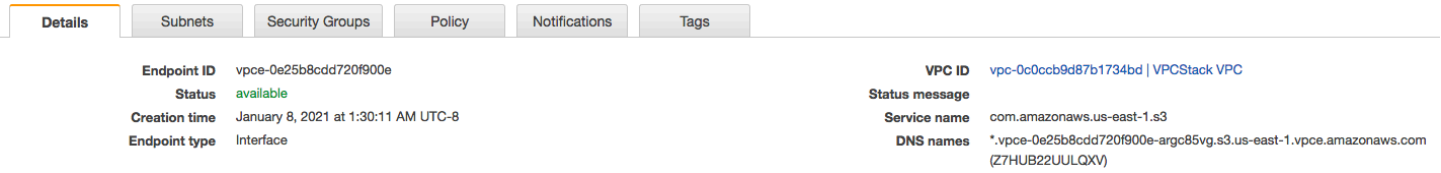
Jika Anda ingin memperbarui titik akhir VPC yang ada untuk mengaktifkan DNS pribadi hanya untuk Inbound Resolver, gunakan contoh berikut dan ganti nilai sampel dengan milik Anda sendiri.

```
aws ec2 modify-vpc-endpoint \
--region us-east-1 \
--vpc-endpoint-id client-vpc-id \
--private-dns-enabled \
--dns-options PrivateDnsOnlyForInboundResolverEndpoint=true
```

Mengakses bucket, titik akses, serta operasi API Amazon S3 Control dari titik akhir antarmuka S3

Anda dapat menggunakan AWS CLI atau AWS SDK untuk mengakses bucket, titik akses S3, dan operasi API Kontrol Amazon S3 melalui titik akhir antarmuka S3.

Citra berikut menunjukkan konsol VPC tab Detail, di mana Anda dapat menemukan nama DNS titik akhir VPC. Dalam contoh ini, ID titik akhir VPC (vpce-id) adalah `vpce-0e25b8cdd720f900e` dan Nama DNS adalah `*.vpce-0e25b8cdd720f900e-argc85vg.s3.us-east-1.vpce.amazonaws.com`.



Saat menggunakan nama DNS untuk mengakses sumber daya, *ganti** dengan nilai yang sesuai. Nilai yang sesuai untuk digunakan sebagai pengganti `*` adalah sebagai berikut:

- `bucket`
- `accesspoint`
- `control`

Misalnya, untuk mengakses bucket, gunakan nama DNS seperti ini:

`bucket.vpce-0e25b8cdd720f900e-argc85vg.s3.us-east-1.vpce.amazonaws.com`

Untuk contoh cara menggunakan nama DNS untuk mengakses bucket, titik akses, dan operasi API Kontrol Amazon S3, lihat bagian berikut dari AWS CLI contoh dan AWS Contoh SDK.

Untuk informasi selengkapnya tentang cara melihat nama DNS kustom titik akhir Anda, lihat Melihat konfigurasi nama DNS privat layanan titik akhir dalam Panduan Pengguna VPC.

AWS CLI contoh

Untuk mengakses bucket S3, titik akses S3, atau operasi API Kontrol Amazon S3 melalui titik akhir antarmuka S3 dalam AWS CLI perintah, gunakan dan parameter. `--region` `--endpoint-url`

Contoh: Gunakan URL titik akhir untuk daftar objek dalam bucket Anda

Pada contoh berikut, ganti nama bucket *my-bucket*, Region *us-east-1*, dan nama DNS *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* ID titik akhir VPC dengan informasi Anda sendiri.

```
aws s3 ls s3://my-bucket/ --region us-east-1 --endpoint-url
 https://bucket.vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com
```

Contoh: Gunakan URL titik akhir untuk membuat daftar objek dari titik akses

- Metode 1—Menggunakan Amazon Resource Name (ARN) dari titik akses dengan titik akhir titik akses

Ganti ARN *us-east-1:123456789012:accesspoint/accesspointexamplename*, Wilayah *us-east-1*, dan ID *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* titik akhir VPC dengan informasi Anda sendiri.

```
aws s3api list-objects-v2 --bucket arn:aws:s3:us-east-1:123456789012:accesspoint/
accesspointexamplename --region us-east-1 --endpoint-url
 https://accesspoint.vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com
```

Jika Anda tidak berhasil menjalankan perintah, perbarui AWS CLI ke versi terbaru dan coba lagi. Untuk informasi selengkapnya tentang petunjuk pemutsufiks, lihat Menginstal atau memperbarui versi terbaru AWS CLI dalam AWS Command Line Interface Panduan Pengguna.

- Metode 2—Menggunakan alias titik akses dengan titik akhir bucket regional

  Dalam contoh berikut, ganti alias titik akses *accesspointexamplename-8tyekmigicmhun8n9kwpfur39dnw4use1a-s3alias*, Wilayah *us-east-1*, dan *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* ID titik akhir VPC dengan informasi Anda sendiri.

```
aws s3api list-objects-v2 --
bucket accesspointexamplename-8tyekmigicmhun8n9kwpfur39dnw4use1a-s3alias
 --region us-east-1 --endpoint-url https://bucket.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com
```

- Metode 3—Menggunakan alias titik akses dengan titik akhir titik akses

  Pertama, untuk membuat titik akhir S3 dengan bucket yang disertakan sebagai bagian dari nama host, atur gaya pengalamatan `virtual` `aws s3api` untuk digunakan. Untuk informasi selengkapnya`AWS configure`, lihat Pengaturan konfigurasi dan file kredensial di Panduan AWS Command Line Interface Pengguna.

```
aws configure set default.s3.addressing_style virtual
```

  Kemudian, dalam contoh berikut, ganti alias titik akses *accesspointexamplename-8tyekmigicmhun8n9kwpfur39dnw4use1a-s3alias*, Wilayah *us-east-1*, dan *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* ID titik akhir VPC dengan informasi Anda sendiri. Untuk informasi selengkapnya tentang alias titik akses, lihatMenggunakan alias gaya bucket untuk titik akses bucket S3 Anda.

```
aws s3api list-objects-v2 --
bucket accesspointexamplename-8tyekmigicmhun8n9kwpfur39dnw4use1a-s3alias --
region us-east-1 --endpoint-url https://accesspoint.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com
```

Contoh: Gunakan URL titik akhir untuk mendaftarkan tugas dengan operasi API kontrol S3

Dalam contoh berikut, ganti Wilayah *us-east-1*, ID titik akhir VPC *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com*, dan ID akun *12345678* dengan informasi Anda sendiri.

```
aws s3control --region us-east-1 --endpoint-url
 https://control.vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com list-jobs --
account-id 12345678
```

AWS Contoh SDK

Untuk mengakses bucket S3, titik akses S3, atau operasi API Kontrol Amazon S3 melalui titik akhir antarmuka S3 saat menggunakan AWS SDK, perbarui SDK Anda ke versi terbaru. Kemudian konfigurasikan klien Anda untuk menggunakan URL titik akhir untuk mengakses bucket, titik akses, atau operasi API Kontrol Amazon S3 melalui titik akhir antarmuka S3.

SDK for Python (Boto3)

Contoh: Gunakan URL titik akhir untuk mengakses bucket S3

Dalam contoh berikut, ganti Wilayah *us-east-1* dan ID titik akhir VPC *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* dengan informasi Anda sendiri.

```
s3_client = session.client(
service_name='s3',
region_name='us-east-1',
endpoint_url='https://bucket.vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com'
)
```

Contoh: Gunakan URL titik akhir untuk mengakses titik akses bucket S3

Dalam contoh berikut, ganti Wilayah *us-east-1* dan ID titik akhir VPC *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* dengan informasi Anda sendiri.

```
ap_client = session.client(
service_name='s3',
region_name='us-east-1',
endpoint_url='https://accesspoint.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com'
)
```

Contoh: Gunakan URL titik akhir untuk mengakses API Kontrol Amazon S3

Dalam contoh berikut, ganti Wilayah *us-east-1* dan ID titik akhir VPC *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* dengan informasi Anda sendiri.

```
control_client = session.client(
service_name='s3control',
region_name='us-east-1',
endpoint_url='https://control.vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com'
)
```

SDK for Java 1.x

Contoh: Gunakan URL titik akhir untuk mengakses bucket S3

Dalam contoh berikut, ganti titik akhir VPC *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* dengan informasi Anda sendiri.

```
// bucket client
final AmazonS3 s3 = AmazonS3ClientBuilder.standard().withEndpointConfiguration(
        new AwsClientBuilder.EndpointConfiguration(
                "https://bucket.vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com",
                Regions.DEFAULT_REGION.getName()
        )
).build();
List<Bucket> buckets = s3.listBuckets();
```

Contoh: Gunakan URL titik akhir untuk mengakses titik akses bucket S3

Dalam contoh berikut, ganti ID titik akhir VPC dan *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* ARN *us-east-1:123456789012:accesspoint/prod* dengan informasi Anda sendiri.

```
// accesspoint client
final AmazonS3 s3accesspoint =
 AmazonS3ClientBuilder.standard().withEndpointConfiguration(
        new AwsClientBuilder.EndpointConfiguration(
                "https://accesspoint.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com",
                Regions.DEFAULT_REGION.getName()
        )
).build();
ObjectListing objects = s3accesspoint.listObjects("arn:aws:s3:us-
east-1:123456789012:accesspoint/prod");
```

Contoh: Gunakan URL titik akhir untuk mengakses operasi API Kontrol Amazon S3

Dalam contoh berikut, ganti titik akhir VPC *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* dengan informasi Anda sendiri.

```
// control client
final AWSS3Control s3control =
 AWSS3ControlClient.builder().withEndpointConfiguration(
        new AwsClientBuilder.EndpointConfiguration(
                "https://control.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com",
                Regions.DEFAULT_REGION.getName()
        )
).build();
final ListJobsResult jobs = s3control.listJobs(new ListJobsRequest());
```

SDK for Java 2.x

Contoh: Gunakan URL titik akhir untuk mengakses bucket S3

Dalam contoh berikut, ganti ID titik akhir VPC *vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com* dan Wilayah *Region.US_EAST_1* dengan informasi Anda sendiri.

```
// bucket client
```

```
Region region = Region.US_EAST_1;
s3Client = S3Client.builder().region(region)

 .endpointOverride(URI.create("https://bucket.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com"))
                    .build()
```

Contoh: Gunakan URL titik akhir untuk mengakses titik akses bucket S3

Dalam contoh berikut, ganti ID titik akhir VPC `vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com` dan Wilayah `Region.US_EAST_1` dengan informasi Anda sendiri.

```
// accesspoint client
Region region = Region.US_EAST_1;
s3Client = S3Client.builder().region(region)

 .endpointOverride(URI.create("https://accesspoint.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com"))
                    .build()
```

Contoh: Gunakan URL titik akhir untuk mengakses API Kontrol Amazon S3

Dalam contoh berikut, ganti ID titik akhir VPC `vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com` dan Wilayah `Region.US_EAST_1` dengan informasi Anda sendiri.

```
// control client
Region region = Region.US_EAST_1;
s3ControlClient = S3ControlClient.builder().region(region)

 .endpointOverride(URI.create("https://control.vpce-1a2b3c4d-5e6f.s3.us-
east-1.vpce.amazonaws.com"))
                                  .build()
```

Memperbarui konfigurasi DNS on-premise

Saat menggunakan nama DNS kustom titik akhir untuk mengakses titik akhir antarmuka untuk Amazon S3, Anda tidak perlu memperbarui penyelesai DNS on-premise Anda. Anda dapat

menyelesaikan nama DNS titik akhir kustom dengan alamat IP privat titik akhir antarmuka dari domain Amazon S3 DNS publik.

Menggunakan titik akhir antarmuka untuk mengakses Amazon S3 tanpa titik akhir gateway atau gateway internet di VPC

Titik akhir antarmuka di VPC Anda dapat merutekan kedua aplikasi di VPC dan aplikasi on-premise ke Amazon S3 melalui jaringan Amazon, seperti yang digambarkan dalam diagram berikut.

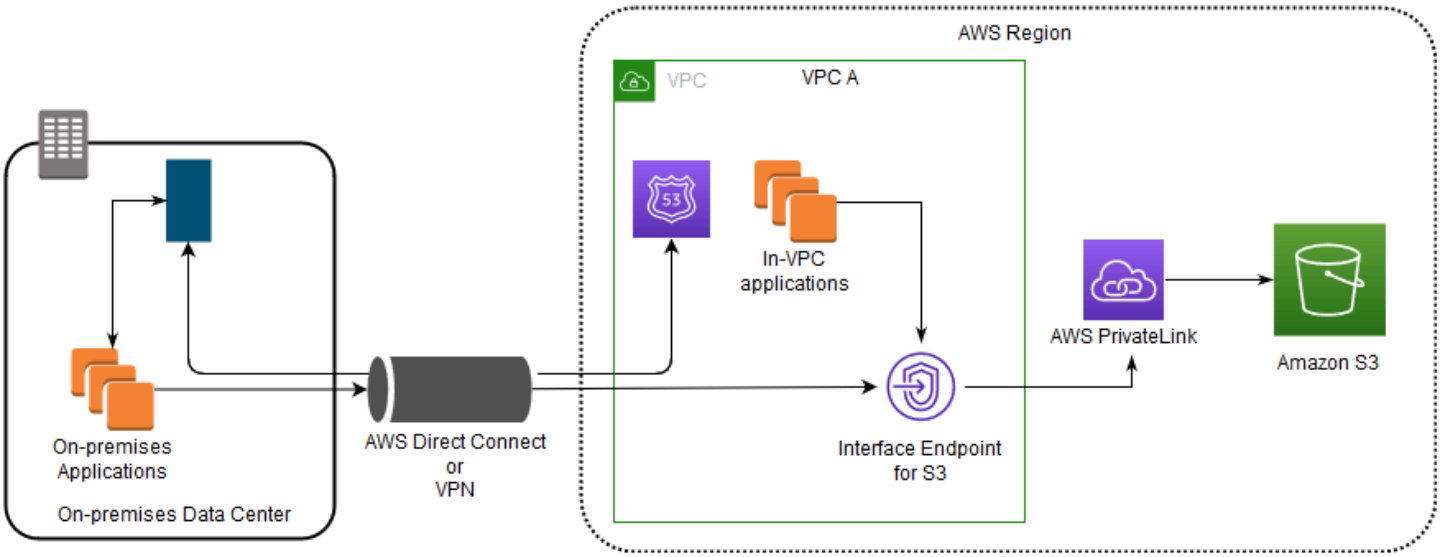


Diagram ini menggambarkan hal sebagai berikut:

- Jaringan lokal Anda menggunakan AWS Direct Connect atau menyambung AWS VPN ke VPC A.
- Aplikasi Anda on-premise dan di VPC A menggunakan nama DNS titik akhir kustom untuk mengakses Amazon S3 melalui titik akhir antarmuka S3.
- Aplikasi lokal mengirim data ke titik akhir antarmuka di VPC melalui AWS Direct Connect (atau). AWS VPN AWS PrivateLink memindahkan data dari titik akhir antarmuka ke Amazon S3 melalui AWS jaringan.
- Aplikasi in-vPC juga mengirim lalu lintas ke titik akhir antarmuka. AWS PrivateLink memindahkan data dari titik akhir antarmuka ke Amazon S3 melalui AWS jaringan.

Menggunakan titik akhir gateway dan antarmuka titik akhir bersama-sama dalam VPC yang sama untuk mengakses Amazon S3

Anda dapat membuat titik akhir antarmuka dan mempertahankan titik akhir gateway yang ada di VPC yang sama, seperti yang ditunjukkan diagram berikut. Dengan mengambil pendekatan ini, Anda

mengizinkan aplikasi dalam VPC untuk terus mengakses Amazon S3 melalui titik akhir gateway, yang tidak ditagih. Kemudian, hanya aplikasi on-premise Anda yang akan menggunakan titik akhir antarmuka untuk mengakses Amazon S3. Untuk mengakses S3 dengan cara ini, Anda harus memperbarui aplikasi lokal Anda untuk menggunakan nama DNS kustom titik akhir untuk Amazon S3.

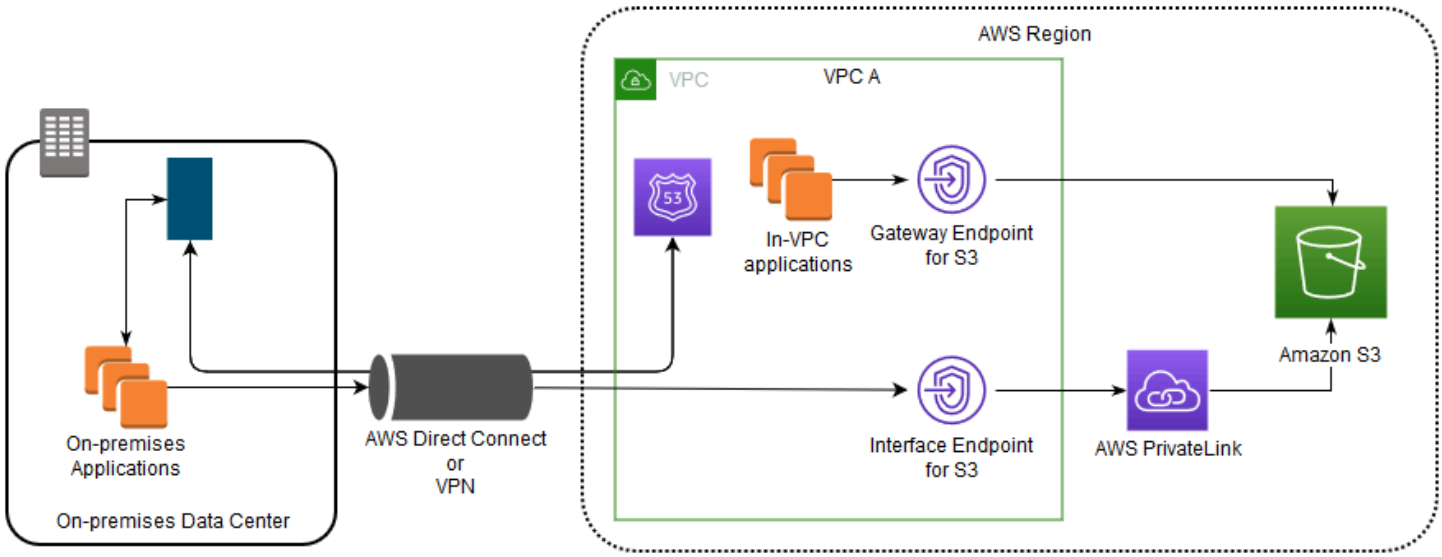


Diagram ini menggambarkan hal sebagai berikut:

- Aplikasi lokal menggunakan nama DNS khusus titik akhir untuk mengirim data ke titik akhir antarmuka dalam VPC melalui (atau). AWS Direct Connect AWS VPN AWS PrivateLink memindahkan data dari titik akhir antarmuka ke Amazon S3 melalui AWS jaringan.
- Menggunakan nama Amazon S3 Regional default, aplikasi in-VPC mengirim data ke titik akhir gateway yang terhubung ke Amazon S3 melalui jaringan. AWS

Untuk informasi selengkapnya tentang titik akhir gateway, lihat Gateway titik akhir VPC di Panduan Pengguna VPC.

Membuat kebijakan titik akhir VPC untuk Amazon S3

Anda dapat melampirkan kebijakan titik akhir ke titik akhir VPC yang mengendalikan akses ke Amazon S3. Kebijakan titik akhir mencantumkan informasi berikut:

- Prinsip AWS Identity and Access Management (IAM) yang dapat melakukan tindakan
- Tindakan-tindakan yang dapat dilakukan
- Sumber daya yang padanya tindakan dapat dilakukan

Anda juga dapat menggunakan kebijakan bucket Amazon S3 untuk membatasi akses ke bucket tertentu dari titik akhir VPC tertentu menggunakan syarat `aws:sourceVpce` dalam kebijakan bucket Anda. Contoh berikut menunjukkan kebijakan yang membatasi akses ke bucket atau titik akhir.

Topik

- Contoh: Membatasi Akses ke bucket kustom dari titik akhir VPC
- Contoh: Membatasi akses ke bucket kustom di akun kustom dari titik akhir VPC
- Contoh: Membatasi Akses ke titik akhir VPC kustom di kebijakan bucket S3

Contoh: Membatasi Akses ke bucket kustom dari titik akhir VPC

Anda dapat membuat kebijakan titik akhir yang membatasi akses ke bucket S3 Amazon tertentu saja. Jenis kebijakan ini berguna jika Anda memiliki kebijakan lain Layanan AWS di VPC yang menggunakan bucket. Kebijakan bucket berikut hanya membatasi akses ke file *example-s3-bucket1*. Untuk menggunakan kebijakan titik akhir ini, ganti *example-s3-bucket1* dengan nama bucket Anda.

```
{
  "Version": "2012-10-17",
  "Id": "Policy1415115909151",
  "Statement": [
    { "Sid": "Access-to-specific-bucket-only",
      "Principal": "*",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::example-s3-bucket1",
                   "arn:aws:s3:::example-s3-bucket1/*"]
    }
  ]
}
```

Contoh: Membatasi akses ke bucket kustom di akun kustom dari titik akhir VPC

Anda dapat membuat kebijakan titik akhir yang membatasi akses hanya ke bucket S3 di bucket tertentu. Akun AWS Untuk mencegah klien dalam VPC Anda mengakses bucket yang tidak Anda miliki, gunakan pernyataan berikut ini di kebijakan titik akhir Anda. Pernyataan contoh berikut

membuat kebijakan yang membatasi akses ke sumber daya yang dimiliki oleh satu Akun AWS ID, *111122223333*.

```
{
  "Statement": [
    {
      "Sid": "Access-to-bucket-in-specific-account-only",
      "Principal": "*",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Deny",
      "Resource": "arn:aws:s3:::*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": "111122223333"
        }
      }
    }
  ]
}
```

> **Note**
>
> Untuk menentukan Akun AWS ID sumber daya yang diakses, Anda dapat menggunakan salah satu `aws:ResourceAccount` atau `s3:ResourceAccount` kunci dalam kebijakan IAM Anda. Namun, ketahuilah bahwa beberapa Layanan AWS mengandalkan akses ke bucket yang AWS dikelola. Oleh karena itu, menggunakan kunci `aws:ResourceAccount` atau `s3:ResourceAccount` dalam kebijakan IAM Anda juga dapat memengaruhi akses ke sumber daya ini.

Contoh: Membatasi Akses ke titik akhir VPC kustom di kebijakan bucket S3

Contoh: Membatasi Akses ke titik akhir VPC kustom di kebijakan bucket S3

Kebijakan bucket Amazon S3 berikut memungkinkan akses ke bucket tertentu, *example-s3-bucket2*, hanya dari titik akhir VPC *vpce-1a2b3c4d*. Kebijakan ini menolak semua akses ke bucket jika titik akhir yang ditentukan tidak digunakan. `aws:sourceVpce`Kondisi menentukan titik akhir dan tidak memerlukan Amazon Resource Name (ARN) untuk sumber daya titik akhir VPC,

tetapi hanya ID titik akhir. Untuk menggunakan kebijakan bucket ini, ganti *example-s3-bucket2* dan *vpce-1a2b3c4d* dengan nama bucket dan titik akhir Anda.

> **Important**
>
> - Saat menerapkan kebijakan bucket Amazon S3 berikut untuk membatasi akses hanya ke titik akhir VPC tertentu, Anda dapat memblokir akses Anda ke bucket tanpa bermaksud melakukannya. Kebijakan bucket yang dimaksudkan untuk secara kustom membatasi akses bucket ke koneksi yang berasal dari titik akhir VPC Anda dapat memblokir semua koneksi ke bucket. Untuk informasi tentang cara mengatasi masalah ini, lihat Kebijakan bucket saya memiliki ID titik akhir VPC atau VPC yang salah. Bagaimana saya dapat memperbaiki kebijakan tersebut sehingga saya dapat mengakses bucket? dalam AWS Support Pusat Pengetahuan.
> - Sebelum menggunakan kebijakan contoh berikut ini, ganti ID titik akhir VPC dengan nilai yang sesuai untuk kasus penggunaan Anda. Jika tidak, Anda tidak akan dapat mengakses bucket Anda.
> - Kebijakan ini nonaktifkan akses konsol ke bucket tertentu, karena permintaan konsol tidak berasal dari titik akhir VPC tertentu.

```
{
  "Version": "2012-10-17",
  "Id": "Policy1415115909152",
  "Statement": [
    { "Sid": "Access-to-specific-VPCE-only",
      "Principal": "*",
      "Action": "s3:*",
      "Effect": "Deny",
      "Resource": ["arn:aws:s3:::example-s3-bucket2",
                   "arn:aws:s3:::example-s3-bucket2/*"],
      "Condition": {"StringNotEquals": {"aws:sourceVpce": "vpce-1a2b3c4d"}}
    }
  ]
}
```

Untuk contoh kebijakan lainnya, lihat Titik akhir untuk Amazon S3 di Panduan Pengguna VPC.

Untuk informasi selengkapnya tentang konektivitas VPC, lihat opsi konektivitas jaringan-ke-VPC di whitepaper Opsi Konektivitas Amazon AWS Virtual Private Cloud.

Manajemen Akses

Di AWS, sumber daya adalah entitas yang dapat Anda gunakan. Di Amazon Simple Storage Service (S3), bucket dan objek adalah sumber daya Amazon S3 asli. Setiap pelanggan S3 kemungkinan memiliki ember dengan benda-benda di dalamnya. Karena fitur baru ditambahkan ke S3, sumber daya tambahan juga ditambahkan, tetapi tidak setiap pelanggan menggunakan sumber daya khusus fitur ini. Untuk informasi selengkapnya tentang sumber daya Amazon S3, lihat. Sumber daya S3

Secara default, semua sumber daya Amazon S3 bersifat pribadi. Secara default, pengguna root dari Akun AWS yang menciptakan sumber daya (pemilik sumber daya) dan pengguna IAM dalam akun itu dengan izin yang diperlukan dapat mengakses sumber daya yang mereka buat. Pemilik sumber daya memutuskan siapa lagi yang dapat mengakses sumber daya dan tindakan yang diizinkan dilakukan orang lain pada sumber daya. S3 memiliki berbagai alat manajemen akses yang dapat Anda gunakan untuk memberi orang lain akses ke sumber daya S3 Anda.

Bagian berikut memberi Anda gambaran umum tentang sumber daya S3, alat manajemen akses S3 yang tersedia, dan kasus penggunaan terbaik untuk setiap alat manajemen akses. Daftar di bagian ini bertujuan untuk menjadi komprehensif dan mencakup semua sumber daya S3, alat manajemen akses, dan kasus penggunaan manajemen akses umum. Pada saat yang sama, bagian ini dirancang untuk menjadi direktori yang mengarahkan Anda ke detail teknis yang Anda inginkan. Jika Anda memiliki pemahaman yang baik tentang beberapa topik berikut, Anda dapat melompat ke bagian yang berlaku untuk Anda.

Topik

- Sumber daya S3
- Identitas
- Alat manajemen akses
- Tindakan
- Kasus penggunaan manajemen akses
- Pemecahan masalah manajemen akses
- Identity and Access Management untuk Amazon S3
- Mengelola akses dengan S3 Access Grants
- Mengelola Akses dengan ACL
- Melakukan blok akses publik ke penyimpanan Amazon S3 Anda
- Meninjau akses bucket menggunakan IAM Access Analyzer untuk S3

- Memverifikasi kepemilikan bucket dengan syarat pemilik bucket
- Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda

Sumber daya S3

Sumber daya Amazon S3 asli adalah ember dan objek yang dikandungnya. Karena fitur baru ditambahkan ke S3, sumber daya baru juga ditambahkan. Berikut ini adalah daftar lengkap sumber daya S3 dan fitur-fiturnya masing-masing.

Jenis sumber daya	Fitur Amazon S3	Deskripsi
`bucket`	Fitur inti	Bucket adalah kontainer untuk objek. Untuk menyimpan objek di S3, buat bucket lalu unggah satu atau beberapa objek ke bucket. Untuk informasi selengkapnya, lihat Membuat, mengonfigurasi, dan bekerja dengan bucket Amazon S3.
`object`		Objek dapat berupa file dan metadata apa pun yang menggambarkan file itu. Ketika sebuah objek ada di ember, Anda dapat membukanya, mengunduhnya, dan memindahk annya. Untuk informasi selengkapnya, lihat Mengunggah, mengunduh, dan bekerja dengan objek di Amazon S3.
`accesspoi nt`	Titik Akses	Titik Akses diberi nama titik akhir jaringan yang dilampirk an ke bucket yang dapat Anda gunakan untuk melakukan operasi objek Amazon S3, seperti dan. `GetObject` `PutObject` Setiap titik akses memiliki izin berbeda, kontrol jaringan, dan kebijakan jalur akses khusus yang berfungsi bersama dengan kebijakan bucket yang dilampirk an ke bucket yang mendasarinya. Anda dapat mengonfig urasi titik akses apa pun untuk menerima permintaan hanya dari virtual private cloud (VPC) atau mengonfigurasi pengaturan akses publik blok khusus untuk setiap titik akses. Untuk informasi selengkapnya, lihat Mengelola akses data dengan titik akses Amazon S3.

Jenis sumber daya	Fitur Amazon S3	Deskripsi
`objectlambdaaccesspoint`		Object Lambda Access Point adalah titik akses untuk bucket yang juga terkait dengan fungsi Lambda. Dengan Object Lambda Access Point, Anda dapat menambahkan kode Anda sendiri ke Amazon `GET` S3`LIST`, `HEAD` dan meminta untuk memodifikasi dan memproses data saat dikembalikan ke aplikasi. Untuk informasi selengkapnya, lihat Membuat Titik Akses Objek Lambda.
`multiregionaccesspoint`		Titik Akses Multi-Wilayah menyediakan titik akhir global yang dapat digunakan aplikasi untuk memenuhi permintaan dari bucket Amazon S3 yang berlokasi di beberapa Wilayah. AWS Anda dapat menggunakan Titik Akses Multi-Wilayah untuk membangun aplikasi multi-Wilayah dengan arsitektur yang sama dengan yang digunakan di satu Wilayah, dan kemudian menjalankan aplikasi tersebut di mana saja di seluruh dunia. Alih-alih mengirim permintaan melalui internet publik yang padat, permintaan aplikasi yang dibuat ke titik akhir global Multi-Region Access Point secara otomatis merutekan melalui jaringan AWS global ke bucket Amazon S3 terdekat. Untuk informasi selengkapnya, lihat Titik Akses Multi-Wilayah di Amazon S3.
`job`	Operasi Batch S3	Pekerjaan adalah sumber daya dari fitur Operasi Batch S3. Anda dapat menggunakan Operasi Batch S3 untuk melakukan operasi batch skala besar pada daftar objek Amazon S3 yang Anda tentukan. Amazon S3 melacak kemajuan pekerjaan operasi batch, mengirimkan pemberitahuan, dan menyimpan laporan penyelesaian terperinci dari semua tindakan, memberi Anda pengalaman yang sepenuhnya dikelola, dapat diaudit, dan tanpa server. Untuk informasi selengkapnya, lihat Melakukan operasi batch berskala besar pada objek Amazon S3.

Jenis sumber daya	Fitur Amazon S3	Deskripsi
`storagelensconfiguration`	Lensa Penyimpanan S3	Konfigurasi Lensa Penyimpanan S3 mengumpulkan metrik penyimpanan di seluruh organisasi dan data pengguna di seluruh akun. S3 Storage Lens memberi admin satu tampilan penggunaan dan aktivitas penyimpanan objek di ratusan, atau bahkan ribuan, akun dalam suatu organisasi, dengan detail untuk menghasilkan wawasan di berbagai tingkat agregasi. Untuk informasi selengkapnya, lihat Menilai aktivitas penyimpanan dan penggunaan Anda dengan Amazon S3 Storage Lens.
`storagelensgroup`		Grup Lensa Penyimpanan S3 menggabungkan metrik dengan menggunakan filter khusus berdasarkan metadata objek. Grup Lensa Penyimpanan S3 membantu Anda menyelidiki karakteristik data Anda, seperti distribusi objek berdasarkan usia, jenis file yang paling umum, dan banyak lagi. Untuk informasi selengkapnya, lihat Bekerja dengan grup Lensa Penyimpanan.
`accessgrantsinstance`	Pemberian Akses S3	Instance S3 Access Grants adalah wadah untuk hibah S3 yang Anda buat. Dengan S3 Access Grants, Anda dapat membuat hibah ke data Amazon S3 untuk identitas IAM dalam akun Anda, identitas IAM di akun lain (lintas akun), dan identitas direktori yang ditambahkan dari direktori perusahaan Anda. AWS IAM Identity Center Untuk informasi selengkapnya tentang Hibah Akses S3, lihat. Mengelola akses dengan S3 Access Grants

Jenis sumber daya	Fitur Amazon S3	Deskripsi
`accessgrantslocation`		Lokasi Access Grants adalah bucket, awalan dalam bucket, atau objek yang Anda daftarkan di instance S3 Access Grants. Anda harus mendaftarkan lokasi dalam instance S3 Access Grants sebelum Anda dapat membuat hibah ke lokasi tersebut. Kemudian, dengan S3 Access Grants, Anda dapat memberikan akses ke bucket, awalan, atau objek untuk identitas IAM dalam akun Anda, identitas IAM di akun lain (lintas akun), dan identitas direktori yang ditambahkan dari direktori perusahaan Anda. AWS IAM Identity Center Untuk informasi selengkapnya tentang Hibah Akses S3, lihat Mengelola akses dengan S3 Access Grants
`accessgrant`		Hibah Akses adalah hibah individu untuk data Amazon S3 Anda. Dengan S3 Access Grants, Anda dapat membuat hibah ke data Amazon S3 untuk identitas IAM dalam akun Anda, identitas IAM di akun lain (lintas akun), dan identitas direktori yang ditambahkan dari direktori perusahaan Anda. AWS IAM Identity Center Untuk informasi selengkapnya tentang Hibah Akses S3, lihat Mengelola akses dengan S3 Access Grants

Bucket

Ada dua jenis bucket Amazon S3: ember tujuan umum dan ember direktori.

- Bucket tujuan umum adalah jenis bucket S3 asli dan direkomendasikan untuk sebagian besar kasus penggunaan dan pola akses. Bucket tujuan umum juga memungkinkan objek yang disimpan di semua kelas penyimpanan, kecuali S3 Express One Zone. Untuk informasi selengkapnya tentang kelas penyimpanan S3, lihatMenggunakan kelas penyimpanan Amazon S3.
- Bucket direktori menggunakan kelas penyimpanan S3 Express One Zone, yang direkomendasikan jika aplikasi Anda peka terhadap kinerja dan mendapat manfaat dari milidetik dan latensi satu digit. `PUT` `GET` Lihat informasi selengkapnya di Ember direktori, Apa itu S3 Express One Zone?, dan AWS Identity and Access Management (IAM) untuk S3 Express One Zone.

Mengkategorikan sumber daya S3

Amazon S3 menyediakan fitur untuk mengkategorikan dan mengatur sumber daya S3 Anda. Mengkategorikan sumber daya Anda tidak hanya berguna untuk mengaturnya, tetapi Anda juga dapat menetapkan aturan manajemen akses berdasarkan kategori sumber daya. Secara khusus, awalan dan penandaan adalah dua fitur organisasi penyimpanan yang dapat Anda gunakan saat mengatur izin manajemen akses.

> **Note**
>
> Informasi berikut berlaku untuk ember tujuan umum. Bucket direktori tidak mendukung penandaan, dan mereka memiliki batasan awalan. Untuk informasi selengkapnya, lihat AWS Identity and Access Management (IAM) untuk S3 Express One Zone.

- Awalan - Awalan di Amazon S3 adalah string karakter di awal nama kunci objek yang digunakan untuk mengatur objek yang disimpan di bucket S3 Anda. Anda dapat menggunakan karakter pembatas, seperti garis miring maju (`/`), untuk menunjukkan akhir awalan dalam nama kunci objek. Misalnya, Anda mungkin memiliki nama kunci objek yang dimulai dengan `engineering/` awalan atau nama kunci objek yang dimulai dengan `marketing/campaigns/` awalan. Menggunakan delimeter di akhir awalan Anda, seperti karakter garis miring maju `/` mengemulasi folder dan konvensi penamaan file. Namun, di S3, awalan adalah bagian dari nama kunci objek. Dalam bucket S3 tujuan umum, tidak ada hierarki folder yang sebenarnya.

  Amazon S3 mendukung pengorganisasian dan pengelompokan objek dengan menggunakan awalan mereka. Anda juga dapat mengelola akses ke objek dengan awalan mereka. Misalnya, Anda dapat membatasi akses hanya ke objek dengan nama yang dimulai dengan awalan tertentu.

  Untuk informasi selengkapnya, lihat Organisasi objek menggunakan prefiks. Konsol S3 menggunakan konsep folder, yang, dalam bucket tujuan umum, pada dasarnya adalah awalan yang telah dituliskan sebelumnya ke nama kunci objek. Untuk informasi selengkapnya, lihat Mengatur objek di konsol Amazon S3 dengan menggunakan folder.
- Tag - Setiap tag adalah pasangan nilai kunci yang Anda tetapkan ke sumber daya. Misalnya, Anda dapat menandai beberapa sumber daya dengan tag`topicCategory=engineering`. Anda dapat menggunakan penandaan untuk membantu alokasi biaya, pengkategorian dan pengorganisasian, dan kontrol akses. Bucket tagging hanya digunakan untuk alokasi biaya. Anda dapat menandai objek, Lensa Penyimpanan S3, pekerjaan, dan Hibah Akses S3 untuk keperluan pengorganisasian atau untuk kontrol akses. Di S3 Access Grants, Anda juga dapat menggunakan penandaan untuk

alokasi biaya. Sebagai contoh mengendalikan akses ke sumber daya dengan menggunakan tag mereka, Anda hanya dapat berbagi objek yang memiliki tag tertentu atau kombinasi tag.

Untuk informasi selengkapnya, lihat Mengontrol akses ke AWS sumber daya menggunakan tag sumber daya di Panduan Pengguna IAM.

Identitas

Di Amazon S3, pemilik sumber daya adalah identitas yang menciptakan sumber daya, seperti ember atau objek. Secara default, hanya pengguna root akun yang membuat sumber daya dan identitas IAM dalam akun yang memiliki izin yang diperlukan dapat mengakses sumber daya S3. Pemilik sumber daya dapat memberikan identitas lain akses ke sumber daya S3 mereka.

Identitas yang tidak memiliki sumber daya dapat meminta akses ke sumber daya tersebut. Permintaan ke sumber daya diautentikasi atau tidak diautentikasi. Permintaan yang diautentikasi harus menyertakan nilai tanda tangan yang mengautentikasi pengirim permintaan, tetapi permintaan yang tidak diautentikasi tidak memerlukan tanda tangan. Kami menyarankan Anda memberikan akses hanya kepada pengguna yang diautentikasi. Untuk informasi lebih lanjut tentang autentikasi permintaan, lihat Membuat permintaan.

> **Important**
>
> Kami menyarankan agar Anda tidak menggunakan kredensyal pengguna Akun AWS root untuk membuat permintaan yang diautentikasi. Sebagai gantinya, buat peran IAM dan berikan akses penuh kepada peran tersebut. Kami merujuk ke pengguna dengan peran ini sebagai pengguna administrator. Anda dapat menggunakan kredensil yang ditetapkan ke peran administrator, bukan kredensil pengguna Akun AWS root, untuk berinteraksi AWS dan melakukan tugas, seperti membuat bucket, membuat pengguna, dan memberikan izin. Untuk informasi selengkapnya, lihat kredensi pengguna Akun AWS root dan kredenal pengguna IAM di Referensi Umum AWS, dan lihat Praktik terbaik keamanan di IAM di Panduan Pengguna IAM.

Identitas mengakses data Anda di Amazon S3 dapat menjadi salah satu dari yang berikut:

Akun AWS pemilik

Akun AWS Yang menciptakan sumber daya. Misalnya, akun yang membuat ember. Akun ini memiliki sumber daya. Untuk informasi selengkapnya, lihat pengguna root AWS akun.

Identitas IAM di akun pemilik yang sama Akun AWS

Saat menyiapkan akun untuk anggota tim baru yang memerlukan akses S3, Akun AWS pemilik dapat menggunakan AWS Identity and Access Management (IAM) untuk membuat pengguna, grup, dan peran. Akun AWS Pemilik kemudian dapat berbagi sumber daya dengan identitas IAM ini. Pemilik akun juga dapat menentukan izin untuk memberikan identitas IAM, yang memungkinkan atau menolak tindakan yang dapat dilakukan pada sumber daya bersama.

Identitas IAM memberikan peningkatan kemampuan, termasuk kemampuan untuk meminta pengguna memasukkan kredensi login sebelum mengakses sumber daya bersama. Dengan menggunakan identitas IAM, Anda dapat menerapkan bentuk otentikasi multi-faktor IAM (MFA) untuk mendukung fondasi identitas yang kuat. Praktik terbaik IAM adalah membuat peran untuk manajemen akses alih-alih memberikan izin kepada setiap pengguna individu. Anda menetapkan pengguna individu ke peran yang sesuai. Untuk informasi selengkapnya, lihat Praktik terbaik keamanan di IAM.

Pemilik AWS akun lain dan identitas IAM mereka (akses lintas akun)

Akun AWS Pemilik juga dapat memberikan pemilik AWS akun lain, atau identitas IAM milik AWS akun lain, akses ke sumber daya.

> **Note**
>
> Delegasi izin — Jika Akun AWS memiliki sumber daya, ia dapat memberikan izin tersebut kepada yang lain. Akun AWS Akun itu kemudian dapat mendelegasikan izin tersebut, atau sebagian dari mereka, kepada pengguna di akun yang sama. Ini disebut sebagai delegasi izin. Tetapi akun yang menerima izin dari akun lain tidak dapat mendelegasikan izin tersebut “lintas akun” ke akun lain. Akun AWS

Pengguna anonim (akses publik)

Akun AWS Pemilik dapat membuat sumber daya publik. Membuat sumber daya publik secara teknis berbagi sumber daya dengan pengguna anonim. Bucket yang dibuat sejak April 2023 memblokir semua akses publik secara default, kecuali jika Anda mengubah pengaturan ini. Kami menyarankan Anda mengatur bucket untuk memblokir akses publik, dan Anda hanya memberikan akses ke pengguna yang diautentikasi. Untuk informasi lebih lanjut tentang pemblokiran akses publik, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

Layanan AWS

Pemilik sumber daya dapat memberikan akses AWS layanan lain ke sumber daya Amazon S3. Misalnya, Anda dapat memberikan `s3:PutObject` izin AWS CloudTrail layanan untuk menulis file log ke bucket Anda. Untuk informasi selengkapnya, lihat Menyediakan akses ke AWS layanan.

Identitas direktori perusahaan

Pemilik sumber daya dapat memberikan pengguna atau peran dari direktori perusahaan Anda akses ke sumber daya S3 dengan menggunakan S3 Access Grants. Untuk informasi selengkapnya tentang menambahkan direktori perusahaan Anda AWS IAM Identity Center, lihat Apa itu Pusat Identitas IAM? .

Pemilik ember atau sumber daya

Akun AWS Yang Anda gunakan untuk membuat bucket dan mengunggah objek memiliki sumber daya tersebut. Pemilik bucket dapat memberikan izin lintas akun kepada orang lain Akun AWS (atau pengguna di akun lain) untuk mengunggah objek.

Ketika pemilik bucket mengizinkan akun lain untuk mengunggah objek ke bucket, pemilik bucket, secara default, memiliki semua objek yang diunggah ke bucket mereka. Namun, jika pengaturan bucket yang dipaksakan oleh pemilik Bucket dan pemilik Bucket dinonaktifkan, Akun AWS yang mengunggah objek memiliki objek tersebut, dan pemilik bucket tidak memiliki izin pada objek yang dimiliki oleh akun lain, dengan pengecualian berikut:

- Pemilik bucket membayar tagihan. Pemilik bucket dapat menolak akses ke objek apa pun, atau menghapus objek apa pun di dalam bucket, tanpa memandang siapa yang memilikinya.
- Pemilik bucket dapat mengarsipkan objek apa pun atau mengembalikan objek yang diarsipkan, terlepas dari siapa yang memilikinya. Pengarsipan mengacu pada kelas penyimpanan yang digunakan untuk menyimpan objek. Untuk informasi selengkapnya, lihat Mengelola siklus hidup penyimpanan Anda.

Alat manajemen akses

Amazon S3 menyediakan beragam fitur dan alat keamanan. Berikut ini adalah daftar lengkap fitur dan alat ini. Anda tidak memerlukan semua alat manajemen akses ini, tetapi Anda harus menggunakan satu atau lebih untuk memberikan akses ke sumber daya Amazon S3 Anda. Aplikasi yang tepat dari alat-alat ini dapat membantu memastikan bahwa sumber daya Anda hanya dapat diakses oleh pengguna yang dituju.

Alat manajemen akses yang paling umum digunakan adalah kebijakan akses. Kebijakan akses dapat berupa kebijakan berbasis sumber daya yang dilampirkan ke AWS sumber daya, seperti kebijakan bucket untuk bucket. Kebijakan akses juga dapat berupa kebijakan berbasis identitas yang dilampirkan pada identitas AWS Identity and Access Management (IAM), seperti pengguna, grup, atau peran IAM. Tulis kebijakan akses untuk memberikan Akun AWS izin kepada pengguna, grup, dan peran IAM untuk melakukan operasi pada sumber daya. Misalnya, Anda dapat memberikan PUT `Object` izin kepada yang lain Akun AWS sehingga akun lain dapat mengunggah objek ke bucket Anda.

Kebijakan akses menjelaskan siapa yang memiliki akses ke hal-hal apa. Saat Amazon S3 menerima permintaan, Amazon S3 harus mengevaluasi semua kebijakan akses untuk menentukan apakah akan mengotorisasi atau menolak permintaan tersebut. Untuk informasi lebih lanjut tentang cara Amazon S3 mengevaluasi kebijakan ini, lihat Bagaimana Amazon S3 Mengotorisasi Permintaan.

Berikut ini adalah alat manajemen akses yang tersedia di Amazon S3.

Kebijakan bucket

Kebijakan bucket Amazon S3 adalah kebijakan berbasis sumber daya berformat JSON AWS Identity and Access Management (IAM) yang dilampirkan ke bucket tertentu. Gunakan kebijakan bucket untuk memberikan izin identitas lain Akun AWS atau IAM untuk bucket dan objek di dalamnya. Banyak kasus penggunaan manajemen akses S3 dapat dipenuhi dengan menggunakan kebijakan bucket. Dengan kebijakan bucket, Anda dapat mempersonalisasi akses bucket untuk membantu memastikan bahwa hanya identitas yang telah Anda setujui yang dapat mengakses sumber daya dan melakukan tindakan di dalamnya. Untuk informasi selengkapnya, lihat Kebijakan bucket untuk Amazon S3.

Berikut ini adalah contoh kebijakan bucket. Anda mengekspresikan kebijakan bucket dengan menggunakan file JSON. Kebijakan contoh ini memberikan izin baca peran IAM ke semua objek di bucket. Ini berisi satu pernyataan bernama`BucketLevelReadPermissions`, yang memungkinkan `s3:GetObject` tindakan (izin baca) pada objek dalam ember bernama`DOC-EXAMPLE-BUCKET1`. Dengan menetapkan peran IAM sebagai`Principal`, kebijakan ini memberikan akses ke setiap pengguna IAM dengan peran ini. Untuk menggunakan kebijakan contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid":"BucketLevelReadPermissions",
```

```
      "Effect":"Allow",
      "Principal": {
     "AWS": "arn:aws:iam::123456789101:role/s3-role"
      },
      "Action":["s3:GetObject"],
      "Resource":["arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"]
    }]
  }
```

> **Note**
>
> Saat membuat kebijakan, hindari penggunaan karakter wildcard (`*`) di elemen `Principal` karena penggunaan karakter wildcard memungkinkan siapa pun mengakses sumber daya Amazon S3 Anda. Sebagai gantinya, secara eksplisit mencantumkan pengguna atau grup yang diizinkan mengakses bucket, atau mencantumkan kondisi yang harus dipenuhi dengan menggunakan klausa kondisi dalam kebijakan. Selain itu, daripada menyertakan karakter wildcard untuk tindakan pengguna atau grup Anda, berikan mereka izin khusus jika berlaku.

Kebijakan berbasis identitas

Kebijakan pengguna berbasis identitas atau IAM adalah jenis kebijakan AWS Identity and Access Management (IAM). Kebijakan berbasis identitas adalah kebijakan berformat JSON yang dilampirkan ke pengguna, grup, atau peran IAM di akun Anda. AWS Anda dapat menggunakan kebijakan berbasis identitas untuk memberikan akses identitas IAM ke bucket atau objek Anda. Anda dapat membuat pengguna, grup, dan peran IAM di akun Anda dan melampirkan kebijakan akses kepada mereka. Anda kemudian dapat memberikan akses ke AWS sumber daya, termasuk sumber daya Amazon S3. Untuk informasi selengkapnya, lihat Kebijakan berbasis identitas untuk Amazon S3.

Berikut ini adalah contoh kebijakan berbasis identitas. Kebijakan contoh memungkinkan peran IAM terkait untuk melakukan enam tindakan (izin) Amazon S3 yang berbeda pada bucket dan objek di dalamnya. Jika Anda melampirkan kebijakan ini ke peran IAM di akun Anda dan menetapkan peran tersebut ke beberapa pengguna IAM Anda, pengguna dengan peran ini akan dapat melakukan tindakan ini pada sumber daya (bucket) yang ditentukan dalam kebijakan Anda. Untuk menggunakan kebijakan contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
  {
    "Sid": "AssignARoleActions",
    "Effect": "Allow",
    "Action": [
   "s3:PutObject",
   "s3:GetObject",
   "s3:ListBucket",
   "s3:DeleteObject",
   "s3:GetBucketLocation"
    ],
    "Resource": [
   "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*",
   "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
    ]
    },
  {
    "Sid": "AssignARoleActions2",
    "Effect": "Allow",
    "Action": "s3:ListAllMyBuckets",
    "Resource": "*"
  }
  ]
}
```

Pemberian Akses S3

Gunakan Hibah Akses S3 untuk membuat hibah akses ke data Amazon S3 Anda untuk kedua identitas di direktori identitas perusahaan, sepertiActive Directory, dan identitas (IAM). AWS Identity and Access Management S3 Access Grants membantu Anda mengelola izin data dalam skala besar. Selain itu, S3 Access Grants mencatat identitas pengguna akhir dan aplikasi yang digunakan untuk mengakses data S3 di. AWS CloudTrail Ini memberikan riwayat audit terperinci hingga identitas pengguna akhir untuk semua akses ke data di bucket S3 Anda. Untuk informasi selengkapnya, lihat Mengelola akses dengan S3 Access Grants.

Titik Akses

Titik Akses Amazon S3 menyederhanakan pengelolaan akses data dalam skala besar untuk aplikasi yang menggunakan kumpulan data bersama di S3. Access Points diberi nama endpoint jaringan yang dilampirkan ke bucket. Anda dapat menggunakan titik akses untuk melakukan operasi objek S3 dalam skala besar, seperti mengunggah dan mengambil objek. Bucket dapat memiliki hingga 10.000 titik akses yang terpasang, dan untuk setiap titik akses, Anda dapat menerapkan izin dan kontrol jaringan yang berbeda untuk memberi Anda kontrol terperinci atas akses ke objek S3 Anda.

Poin Akses S3 dapat dikaitkan dengan bucket di akun yang sama atau di akun tepercaya lainnya. Kebijakan Access Points adalah kebijakan berbasis sumber daya yang dievaluasi bersama dengan kebijakan bucket yang mendasarinya. Untuk informasi selengkapnya, lihat Mengelola akses data dengan titik akses Amazon S3.

Daftar kontrol akses (ACL)

ACL adalah daftar hibah yang mengidentifikasi penerima hibah dan izin yang diberikan. ACL memberikan izin baca atau tulis dasar kepada yang lain. Akun AWS ACL menggunakan skema XML kustom Amazon S3. ACL adalah jenis kebijakan AWS Identity and Access Management (IAM). Objek ACL digunakan untuk mengelola akses ke objek, dan bucket ACL digunakan untuk mengelola akses ke bucket. Dengan kebijakan bucket, ada satu kebijakan untuk seluruh bucket, tetapi ACL objek ditentukan untuk setiap objek. Kami menyarankan agar Anda mematikan ACL, kecuali dalam keadaan yang tidak biasa di mana Anda harus mengontrol akses secara individual untuk setiap objek. Untuk informasi selengkapnya tentang penggunaan ACL, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

> **Warning**
>
> Mayoritas kasus penggunaan modern di Amazon S3 tidak memerlukan penggunaan ACL.

Berikut ini adalah contoh bucket ACL. Hibah di ACL menunjukkan pemilik bucket yang memiliki izin kontrol penuh.

```
<?xml version="1.0" encoding="UTF-8"?>
 <AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Owner>
   <ID>Owner-Canonical-User-ID</ID>
   <DisplayName>owner-display-name</DisplayName>
  </Owner>
 <AccessControlList>
  <Grant>
   <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Canonical
 User">
    <ID>Owner-Canonical-User-ID</ID>
    <DisplayName>display-name</DisplayName>
   </Grantee>
   <Permission>FULL_CONTROL</Permission>
  </Grant>
 </AccessControlList>
```

```
</AccessControlPolicy>
```

Kepemilikan Objek

Untuk mengelola akses ke objek Anda, Anda harus menjadi pemilik objek. Anda dapat menggunakan setelan tingkat ember Kepemilikan Objek untuk mengontrol kepemilikan objek yang diunggah ke bucket. Juga, gunakan Object Ownership untuk mengaktifkan ACL. Secara default, Kepemilikan Objek disetel ke setelan diberlakukan pemilik Bucket dan semua ACL dimatikan. Saat ACL dimatikan, pemilik bucket memiliki semua objek di bucket dan secara eksklusif mengelola akses ke data. Untuk mengelola akses, pemilik bucket menggunakan kebijakan atau alat manajemen akses lainnya, tidak termasuk ACL. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Object Ownership memiliki tiga pengaturan yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket dan mengaktifkan ACL:

ACL dimatikan

- Pemilik bucket diberlakukan (default) — ACL dimatikan, dan pemilik bucket secara otomatis memiliki dan memiliki kontrol penuh atas setiap objek di bucket. ACL tidak memengaruhi izin ke data di bucket S3. Bucket menggunakan kebijakan secara eksklusif untuk menentukan kontrol akses.

ACL dihidupkan

- Pemilik bucket yang dipilih—Pemilik bucket memiliki dan diberikan kendali penuh atas objek baru yang ditulis akun lain ke bucket dengan ACL `bucket-owner-full-control` yang dibatasi.
- Penulis objek — Akun AWS Yang mengunggah objek memiliki objek, memiliki kontrol penuh atasnya, dan dapat memberikan pengguna lain akses ke sana melalui ACL.

Praktik terbaik tambahan

Pertimbangkan untuk menggunakan pengaturan dan alat bucket berikut untuk membantu melindungi data saat transit dan saat istirahat, keduanya sangat penting dalam menjaga integritas dan aksesibilitas data Anda:

- Blokir Akses Publik - Jangan matikan pengaturan tingkat ember default Blokir Akses Publik. Pengaturan ini memblokir akses publik ke data Anda secara default. Untuk informasi lebih lanjut

tentang pemblokiran akses publik, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

- Pembuatan Versi S3 — Untuk integritas data, Anda dapat menerapkan pengaturan bucket S3 Versioning, yang membuat versi objek Anda saat Anda melakukan pembaruan, alih-alih menimpanya. Anda dapat menggunakan S3 Versioning untuk mempertahankan, mengambil, dan memulihkan versi sebelumnya, jika diperlukan. Untuk informasi tentang Penentuan Versi S3, lihat Menggunakan Penentuan Versi dalam bucket S3.
- S3 Object Lock - S3 Object Lock adalah pengaturan lain yang dapat Anda terapkan untuk mencapai integritas data. Fitur ini dapat mengimplementasikan model write-once-read-many (WORM) untuk menyimpan objek secara abadi. Untuk informasi tentang Kunci Objek, lihat Menggunakan Kunci Objek S3.
- Enkripsi objek — Amazon S3 menawarkan beberapa opsi enkripsi objek yang melindungi data saat transit dan saat istirahat. Enkripsi sisi server mengenkripsi objek Anda sebelum menyimpannya di disk di pusat datanya dan kemudian mendekripsi saat Anda mengunduh objek. Jika Anda mengautentikasi permintaan Anda dan Anda memiliki izin akses, tidak ada perbedaan dalam cara Anda mengakses objek terenkripsi atau tidak terenkripsi. Untuk informasi selengkapnya, lihat Melindungi data dengan enkripsi di sisi klien. S3 mengenkripsi objek yang baru diunggah secara default. Untuk informasi selengkapnya, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3. Enkripsi sisi klien adalah tindakan mengenkripsi data sebelum mengirimkannya ke Amazon S3. Untuk informasi selengkapnya, lihat Melindungi data menggunakan enkripsi di sisi klien.
- Metode penandatanganan — Signature Version 4 adalah proses menambahkan informasi otentikasi ke AWS permintaan yang dikirim oleh HTTP. Untuk keamanan, sebagian besar permintaan AWS harus ditandatangani dengan kunci akses, yang terdiri dari ID kunci akses dan kunci akses rahasia. Kedua kunci ini umumnya disebut sebagai kredensial keamanan Anda. Untuk informasi selengkapnya, lihat Permintaan Autentikasi (Tanda Tangan AWS Versi 4) dan Proses penandatanganan Tanda Tangan Versi 4.

Tindakan

Untuk daftar lengkap izin dan kunci kondisi S3, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Tindakan

Tindakan AWS Identity and Access Management (IAM) untuk Amazon S3 adalah kemungkinan tindakan yang dapat dilakukan pada bucket atau objek S3. Anda memberikan tindakan ini ke identitas sehingga mereka dapat bertindak berdasarkan sumber daya S3 Anda. Contoh tindakan S3 adalah `s3:GetObject` membaca objek dalam ember, dan `s3:PutObject` menulis objek ke ember.

Kunci syarat

Selain tindakan, kunci kondisi IAM dibatasi untuk memberikan akses hanya ketika suatu kondisi terpenuhi. Kunci kondisi bersifat opsional.

> **Note**
>
> Dalam kebijakan akses berbasis sumber daya, seperti kebijakan bucket, atau dalam kebijakan berbasis identitas, Anda dapat menentukan hal berikut:
>
> - Tindakan atau serangkaian tindakan dalam `Action` elemen pernyataan kebijakan.
> - Dalam `Effect` elemen pernyataan kebijakan, Anda dapat menentukan `Allow` untuk memberikan tindakan yang tercantum, atau Anda dapat menentukan `Deny` untuk memblokir tindakan yang tercantum. Untuk lebih mempertahankan praktik hak istimewa terkecil, `Deny` pernyataan dalam `Effect` elemen kebijakan akses harus seluas mungkin, dan `Allow` pernyataan harus sesempit mungkin. `Deny`efek yang dipasangkan dengan `s3:*` tindakan adalah cara lain yang baik untuk menerapkan praktik terbaik opt-in untuk identitas yang termasuk dalam pernyataan kondisi kebijakan.
> - Kunci kondisi dalam `Condition` elemen pernyataan kebijakan.

Kasus penggunaan manajemen akses

Amazon S3 menyediakan pemilik sumber daya dengan berbagai alat untuk memberikan akses. Alat manajemen akses S3 yang Anda gunakan bergantung pada sumber daya S3 yang ingin Anda bagikan, identitas yang Anda berikan akses, dan tindakan yang ingin Anda izinkan atau tolak. Anda mungkin ingin menggunakan satu atau kombinasi alat manajemen akses S3 untuk mengelola akses ke sumber daya S3 Anda.

Dalam kebanyakan kasus, Anda dapat menggunakan kebijakan akses untuk mengelola izin. Kebijakan akses dapat berupa kebijakan berbasis sumber daya, yang dilampirkan ke sumber daya, seperti bucket, atau sumber daya Amazon S3 lainnya (). Sumber daya S3 Kebijakan akses

juga dapat berupa kebijakan berbasis identitas, yang dilampirkan ke pengguna, grup, atau peran AWS Identity and Access Management (IAM) di akun Anda. Anda mungkin menemukan bahwa kebijakan bucket berfungsi lebih baik untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat Kebijakan bucket untuk Amazon S3. Atau, dengan AWS Identity and Access Management (IAM), Anda dapat membuat pengguna, grup, dan peran IAM dalam diri Anda Akun AWS dan mengelola akses mereka ke bucket dan objek melalui kebijakan berbasis identitas. Untuk informasi selengkapnya, lihat Kebijakan berbasis identitas untuk Amazon S3.

Untuk membantu Anda menavigasi opsi manajemen akses ini, berikut ini adalah kasus penggunaan pelanggan Amazon S3 yang umum dan rekomendasi untuk masing-masing alat manajemen akses S3.

Akun AWS Pemilik ingin berbagi bucket hanya dengan pengguna dalam akun yang sama

Semua alat manajemen akses dapat memenuhi kasus penggunaan dasar ini. Kami merekomendasikan alat manajemen akses berikut untuk kasus penggunaan ini:

- Kebijakan bucket — Jika Anda ingin memberikan akses ke satu bucket atau sejumlah kecil bucket, atau jika izin akses bucket Anda serupa dari bucket ke bucket, gunakan kebijakan bucket. Dengan kebijakan bucket, Anda mengelola satu kebijakan untuk setiap bucket. Untuk informasi selengkapnya, lihat Kebijakan bucket untuk Amazon S3.
- Kebijakan berbasis identitas — Jika Anda memiliki jumlah bucket yang sangat besar dengan izin akses berbeda untuk setiap bucket, dan hanya beberapa peran pengguna yang harus dikelola, Anda dapat menggunakan kebijakan IAM untuk pengguna, grup, atau peran. Kebijakan IAM juga merupakan pilihan yang baik jika Anda mengelola akses pengguna ke AWS sumber daya lain, serta sumber daya Amazon S3. Untuk informasi selengkapnya, lihat Contoh 1: Pemilik bucket yang memberikan izin bucket kepada penggunanya.
- Hibah Akses S3 — Anda dapat menggunakan Hibah Akses S3 untuk memberikan akses ke bucket, awalan, atau objek S3 Anda. S3 Access Grants memungkinkan Anda menentukan berbagai izin tingkat objek dalam skala besar; sedangkan, kebijakan bucket dibatasi hingga 20 KB. Untuk informasi selengkapnya, lihat Memulai S3 Access Grants.
- Access Points — Anda dapat menggunakan Access Points, yang diberi nama endpoint jaringan yang dilampirkan ke bucket. Bucket dapat memiliki hingga 10.000 titik akses yang terpasang, dan untuk setiap titik akses, Anda dapat menerapkan izin dan kontrol jaringan yang berbeda untuk memberi Anda kontrol terperinci atas akses ke objek S3 Anda. Untuk informasi selengkapnya, lihat Mengelola akses data dengan titik akses Amazon S3.

Akun AWS Pemilik ingin berbagi bucket atau objek dengan pengguna dari AWS akun lain (cross-account)

Untuk memberikan izin kepada orang lain Akun AWS, Anda harus menggunakan kebijakan bucket atau salah satu alat manajemen akses yang disarankan berikut ini. Anda tidak dapat menggunakan kebijakan akses berbasis identitas untuk kasus penggunaan ini. Untuk informasi selengkapnya tentang pemberian akses lintas akun, lihat Bagaimana cara menyediakan akses lintas akun ke objek yang ada di bucket Amazon S3?

Kami merekomendasikan alat manajemen akses berikut untuk kasus penggunaan ini:

- Kebijakan bucket — Dengan kebijakan bucket, Anda mengelola satu kebijakan untuk setiap bucket. Untuk informasi selengkapnya, lihat Kebijakan bucket untuk Amazon S3.
- Hibah Akses S3 — Anda dapat menggunakan Hibah Akses S3 untuk memberikan izin lintas akun ke bucket, awalan, atau objek S3 Anda. Anda dapat menggunakan S3 Access Grants untuk menentukan berbagai izin tingkat objek pada skala; sedangkan, kebijakan bucket dibatasi hingga 20 KB dalam ukuran. Untuk informasi selengkapnya, lihat Memulai S3 Access Grants.
- Access Points — Anda dapat menggunakan Access Points, yang diberi nama endpoint jaringan yang dilampirkan ke bucket. Bucket dapat memiliki hingga 10.000 titik akses yang terpasang, dan untuk setiap titik akses, Anda dapat menerapkan izin dan kontrol jaringan yang berbeda untuk memberi Anda kontrol terperinci atas akses ke objek S3 Anda. Untuk informasi selengkapnya, lihat Mengelola akses data dengan titik akses Amazon S3.

Akun AWS Pemilik atau pemilik bucket harus memberikan izin di tingkat objek atau tingkat awalan, dan izin ini bervariasi dari objek ke objek atau awalan ke awalan

Dalam kebijakan bucket, misalnya, Anda dapat memberikan akses ke objek dalam bucket yang membagikan awalan nama kunci tertentu atau memiliki tag tertentu. Anda dapat memberikan izin baca pada objek yang dimulai dengan awalan `logs/` nama kunci. Namun, jika izin akses Anda bervariasi menurut objek, pemberian izin ke objek individual dengan menggunakan kebijakan bucket mungkin tidak praktis, terutama karena kebijakan bucket dibatasi hingga 20 KB.

Kami merekomendasikan alat manajemen akses berikut untuk kasus penggunaan ini:

- Hibah Akses S3 - Anda dapat menggunakan Hibah Akses S3 untuk mengelola izin tingkat objek atau tingkat awalan. Tidak seperti kebijakan bucket, Anda dapat menggunakan S3 Access Grants untuk menentukan berbagai izin tingkat objek dalam skala besar. Kebijakan bucket dibatasi hingga ukuran 20 KB. Untuk informasi selengkapnya, lihat Memulai S3 Access Grants.

- Access Points — Anda dapat menggunakan titik akses untuk mengelola izin tingkat objek atau tingkat awalan. Access Points diberi nama endpoint jaringan yang dilampirkan ke bucket. Bucket dapat memiliki hingga 10.000 titik akses yang terpasang, dan untuk setiap titik akses, Anda dapat menerapkan izin dan kontrol jaringan yang berbeda untuk memberi Anda kontrol terperinci atas akses ke objek S3 Anda. Untuk informasi selengkapnya, lihat Mengelola akses data dengan titik akses Amazon S3.
- ACL — Kami tidak menyarankan menggunakan Access Control Lists (ACL), terutama karena ACL dibatasi hingga 100 hibah per objek. Namun, jika Anda memilih untuk mengaktifkan ACL, di Pengaturan Bucket, setel Kepemilikan Objek ke pilihan pemilik Bucket dan ACL diaktifkan. Dengan pengaturan ini, objek baru yang ditulis dengan ACL terekam `bucket-owner-full-control` secara otomatis dimiliki oleh pemilik bucket daripada penulis objek. Anda kemudian dapat menggunakan ACL objek, yang merupakan kebijakan akses berformat XML, untuk memberikan pengguna lain akses ke objek. Untuk informasi selengkapnya, lihat Gambaran umum daftar kontrol akses (ACL).

Akun AWS Pemilik atau pemilik bucket ingin membatasi akses bucket hanya ke ID akun tertentu

Kami merekomendasikan alat manajemen akses berikut untuk kasus penggunaan ini:

- Kebijakan bucket — Dengan kebijakan bucket, Anda mengelola satu kebijakan untuk setiap bucket. Untuk informasi selengkapnya, lihat Kebijakan bucket untuk Amazon S3.
- Access Points — Access Points diberi nama endpoint jaringan yang dilampirkan ke bucket. Bucket dapat memiliki hingga 10.000 titik akses yang terpasang, dan untuk setiap titik akses, Anda dapat menerapkan izin dan kontrol jaringan yang berbeda untuk memberi Anda kontrol terperinci atas akses ke objek S3 Anda. Untuk informasi selengkapnya, lihat Mengelola akses data dengan titik akses Amazon S3.

Akun AWS Pemilik atau pemilik bucket menginginkan titik akhir yang berbeda untuk setiap pengguna atau aplikasi yang mengakses data mereka

Kami merekomendasikan alat manajemen akses berikut untuk kasus penggunaan ini:

- Access Points — Access Points diberi nama endpoint jaringan yang dilampirkan ke bucket. Bucket dapat memiliki hingga 10.000 titik akses yang terpasang, dan untuk setiap titik akses, Anda dapat menerapkan izin dan kontrol jaringan yang berbeda untuk memberi Anda kontrol terperinci atas akses ke objek S3 Anda. Setiap titik akses memberlakukan kebijakan titik akses

khusus yang bekerja bersama kebijakan bucket yang melekat pada bucket dasar. Untuk informasi selengkapnya, lihat Mengelola akses data dengan titik akses Amazon S3.

Akun AWS Pemilik atau pemilik bucket harus mengelola akses dari titik akhir Virtual Private Cloud (VPC) untuk S3

Titik akhir Virtual Private Cloud (VPC) untuk Amazon S3 adalah entitas logis dalam VPC yang memungkinkan konektivitas hanya ke S3. Kami merekomendasikan alat manajemen akses berikut untuk kasus penggunaan ini:

- Bucket dalam pengaturan VPC — Anda dapat menggunakan kebijakan bucket untuk mengontrol siapa yang diizinkan mengakses bucket Anda dan titik akhir VPC mana yang dapat mereka akses. Untuk informasi selengkapnya, lihat Megendalikan akses dari titik akhir VPC dengan kebijakan bucket.
- Access Points — Jika Anda memilih untuk mengatur jalur akses, Anda dapat menggunakan kebijakan titik akses. Anda dapat mengonfigurasi titik akses apa pun untuk menerima permintaan hanya dari cloud privat virtual (VPC) untuk membatasi akses data Amazon S3 ke jaringan pribadi. Anda juga dapat mengonfigurasi pengaturan blok akses publik khusus untuk setiap titik akses. Untuk informasi selengkapnya, lihat Mengelola akses data dengan titik akses Amazon S3.

Akun AWS Pemilik atau pemilik bucket harus membuat situs web statis tersedia untuk umum

Dengan S3, Anda dapat meng-host situs web statis dan memungkinkan siapa saja untuk melihat konten situs web, yang di-host dari ember S3.

Kami merekomendasikan alat manajemen akses berikut untuk kasus penggunaan ini:

- Amazon CloudFront - Solusi ini memungkinkan Anda untuk meng-host situs web statis Amazon S3 ke publik sambil juga terus memblokir semua akses publik ke konten bucket. Jika Anda ingin mengaktifkan keempat pengaturan Akses Publik Blok S3 dan meng-host situs web statis S3, Anda dapat menggunakan Amazon CloudFront Origin Access Control (OAC). Amazon CloudFront menyediakan kemampuan yang diperlukan untuk menyiapkan situs web statis yang aman. Selain itu, situs web statis Amazon S3 yang tidak menggunakan solusi ini hanya dapat mendukung titik akhir HTTP. CloudFront menggunakan penyimpanan Amazon S3 yang tahan lama sambil menyediakan header keamanan tambahan, seperti HTTPS. HTTPS menambahkan keamanan dengan mengenkripsi permintaan HTTP normal, dan melindungi dari serangan siber umum.

Untuk informasi selengkapnya, lihat Memulai situs web statis aman di Panduan CloudFront Pengembang Amazon.

- Membuat bucket Amazon S3 Anda dapat diakses publik — Anda dapat mengonfigurasi bucket untuk digunakan sebagai situs web statis yang diakses publik.

> **Warning**
>
> Kami tidak merekomendasikan metode ini. Sebagai gantinya, kami sarankan Anda menggunakan situs web statis Amazon S3 sebagai bagian dari Amazon. CloudFront Untuk informasi selengkapnya, lihat opsi sebelumnya, atau lihat Memulai situs web statis yang aman.

Untuk membuat situs web statis Amazon S3, tanpa Amazon CloudFront, pertama, Anda harus mematikan semua pengaturan Blokir Akses Publik. Saat menulis kebijakan bucket untuk situs web statis, pastikan Anda hanya mengizinkan tindakan `s3:GetObject`, bukan izin `ListObject` atau `PutObject`. Ini membantu memastikan bahwa pengguna tidak dapat melihat semua objek di bucket Anda atau menambahkan konten mereka sendiri. Untuk informasi selengkapnya, lihat Mengatur izin untuk akses situs web.

Akun AWS Pemilik atau pemilik ember ingin membuat konten ember tersedia untuk umum

Saat membuat bucket Amazon S3 baru, pengaturan Blokir Akses Publik diaktifkan secara default. Untuk informasi lebih lanjut tentang pemblokiran akses publik, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

Kami tidak menyarankan untuk mengizinkan akses publik ke bucket Anda. Namun, jika Anda harus melakukannya untuk kasus penggunaan tertentu, kami merekomendasikan alat manajemen akses berikut untuk kasus penggunaan ini:

- Nonaktifkan pengaturan Blokir Akses Publik — Pemilik bucket dapat mengizinkan permintaan yang tidak diautentikasi ke bucket. Misalnya, permintaan Objek PUT yang tidak diautentikasi diizinkan saat bucket memiliki kebijakan bucket publik, atau saat bucket ACL memberikan akses publik. Semua permintaan yang tidak diautentikasi dibuat oleh AWS pengguna arbitrer lainnya, atau bahkan pengguna anonim yang tidak diautentikasi. Pengguna ini diwakili dalam ACL oleh `65a011a29cdf8ec533ec3d1ccaae921c` ID pengguna kanonik tertentu. Jika objek diunggah ke `WRITE` atau`FULL_CONTROL`, maka ini secara khusus memberikan akses ke grup Semua Pengguna

atau pengguna anonim. Untuk informasi lebih lanjut tentang kebijakan bucket publik dan daftar kontrol akses (ACL) publik, lihat Arti “publik”.

Akun AWS Pemilik atau pemilik bucket telah melampaui batas ukuran kebijakan akses

Kebijakan bucket dan kebijakan berbasis identitas memiliki batas ukuran 20 KB. Jika persyaratan izin akses Anda rumit, Anda mungkin melebihi batas ukuran ini.

Kami merekomendasikan alat manajemen akses berikut untuk kasus penggunaan ini:

- Titik Akses — Gunakan titik akses jika ini berfungsi dengan kasus penggunaan Anda. Dengan titik akses, setiap bucket memiliki beberapa titik akhir jaringan bernama, masing-masing dengan kebijakan jalur aksesnya sendiri yang sesuai dengan kebijakan bucket yang mendasarinya. Namun, titik akses hanya dapat bertindak pada objek, bukan ember, dan tidak mendukung replikasi lintas wilayah. Untuk informasi selengkapnya, lihat Mengelola akses data dengan titik akses Amazon S3.
- Hibah Akses S3 — Gunakan Hibah Akses S3, yang mendukung sejumlah besar hibah yang memberikan akses ke bucket, awalan, atau objek. Untuk informasi selengkapnya, lihat Memulai S3 Access Grants.

Peran Akun AWS pemilik atau admin ingin memberikan akses bucket, awalan, atau objek langsung ke pengguna atau grup di direktori perusahaan

Alih-alih mengelola pengguna, grup, dan peran melalui AWS Identity and Access Management (IAM), Anda dapat menambahkan direktori perusahaan Anda ke AWS IAM Identity Center. Untuk informasi lebih lanjut, lihat Apa itu Pusat Identitas IAM? .

Setelah Anda menambahkan direktori perusahaan Anda AWS IAM Identity Center, kami sarankan Anda menggunakan alat manajemen akses berikut untuk memberikan identitas direktori perusahaan akses ke sumber daya S3 Anda:

- Hibah Akses S3 — Gunakan Hibah Akses S3, yang mendukung pemberian akses ke pengguna atau peran di direktori perusahaan Anda. Untuk informasi selengkapnya, lihat Memulai S3 Access Grants.

Akun AWS Pemilik atau pemilik bucket ingin memberikan akses AWS CloudFront layanan untuk menulis CloudFront log ke bucket S3

Kami merekomendasikan alat manajemen akses berikut untuk kasus penggunaan ini:

- Bucket ACL — Satu-satunya kasus penggunaan yang disarankan untuk ACL bucket adalah memberikan izin kepada tertentu Layanan AWS, seperti akun Amazon. CloudFront `awslogsdelivery` Saat Anda membuat atau memperbarui distribusi dan mengaktifkan CloudFront logging, CloudFront perbarui bucket ACL untuk memberikan `FULL_CONTROL` izin `awslogsdelivery` akun untuk menulis log ke bucket Anda. Untuk informasi selengkapnya, lihat Izin yang diperlukan untuk mengonfigurasi pencatatan standar dan mengakses file log Anda di Panduan CloudFront Pengembang Amazon. Jika bucket yang menyimpan log menggunakan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek S3 untuk mematikan ACL, CloudFront tidak dapat menulis log ke bucket. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Anda, sebagai pemilik bucket, ingin mempertahankan kontrol penuh atas objek yang ditambahkan ke bucket oleh pengguna lain

Anda dapat memberikan akses akun lain untuk mengunggah objek ke bucket menggunakan kebijakan bucket, titik akses, atau S3 Access Grants. Jika Anda telah memberikan akses lintas akun ke bucket Anda, Anda dapat memastikan bahwa objek apa pun yang diunggah ke bucket Anda tetap berada di bawah kendali penuh Anda.

Kami merekomendasikan alat manajemen akses berikut untuk kasus penggunaan ini:

- Kepemilikan Objek - Pertahankan pengaturan tingkat ember Kepemilikan Objek pada pengaturan default pemilik Bucket yang diberlakukan.

Pemecahan masalah manajemen akses

Sumber daya berikut dapat membantu Anda memecahkan masalah apa pun dengan manajemen akses S3:

Pemecahan Masalah Kesalahan Akses Ditolak (403 Terlarang)

Jika Anda mengalami masalah penolakan akses, periksa pengaturan tingkat akun dan tingkat ember. Juga, periksa fitur manajemen akses yang Anda gunakan untuk memberikan akses

untuk memastikan bahwa kebijakan, pengaturan, atau konfigurasi sudah benar. Untuk informasi selengkapnya tentang penyebab umum kesalahan Akses Ditolak (403 Terlarang) di Amazon S3, lihat. Pecahkan masalah kesalahan Akses Ditolak (403 Forbidden) di Amazon S3

Penganalisis Akses IAM untuk S3

Jika Anda tidak ingin membuat sumber daya Anda tersedia untuk umum, atau jika Anda ingin membatasi akses publik ke sumber daya Anda, Anda dapat menggunakan IAM Access Analyzer untuk S3. Di konsol Amazon S3, gunakan IAM Access Analyzer untuk S3 untuk meninjau semua bucket yang memiliki daftar kontrol akses bucket (ACL), kebijakan bucket, atau kebijakan titik akses yang memberikan akses publik atau bersama. IAM Access Analyzer for S3 memberi tahu Anda tentang bucket yang dikonfigurasi untuk memungkinkan akses ke siapa pun di internet atau lainnya Akun AWS, termasuk Akun AWS di luar organisasi Anda. Untuk setiap bucket publik atau bucket bersama, Anda akan menerima temuan yang melaporkan sumber dan tingkat akses publik atau akses bersama.

Di IAM Access Analyzer untuk S3, Anda dapat memblokir semua akses publik ke bucket dengan satu tindakan. Kami menyarankan Anda memblokir semua akses publik ke bucket Anda, kecuali Anda memerlukan akses publik untuk mendukung kasus penggunaan tertentu. Sebelum Anda memblokir semua akses publik, pastikan aplikasi Anda akan terus berfungsi dengan benar tanpa akses publik. Untuk informasi selengkapnya, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

Anda juga dapat meninjau pengaturan izin tingkat ember Anda untuk mengonfigurasi tingkat akses yang terperinci. Untuk kasus penggunaan tertentu dan terverifikasi yang memerlukan akses publik atau akses bersama, Anda dapat menyatakan dan mencatat maksud Anda untuk bucket agar tetap sebagai bucket dengan akses publik atau akses bersama dengan mengarsipkan temuan untuk bucket tersebut. Anda dapat mempertahankan dan memodifikasi konfigurasi bucket ini kapan saja. Anda juga dapat mengunduh temuan Anda dalam bentuk laporan CSV untuk keperluan audit.

IAM Access Analyzer untuk S3 tersedia tanpa biaya tambahan, yang berada di konsol Amazon S3. Penganalisis Akses IAM untuk S3 didukung oleh Penganalisis Akses IAM (IAM) AWS Identity and Access Management . Untuk menggunakan IAM Access Analyzer untuk S3 di konsol Amazon S3, Anda harus mengunjungi Konsol IAM dan membuat penganalisis tingkat akun di IAM Access Analyzer untuk setiap Wilayah.

Untuk informasi selengkapnya tentang IAM Access Analyzer untuk Amazon S3, lihat Meninjau akses bucket menggunakan IAM Access Analyzer untuk S3.

Pencatatan dan pemantauan

Pemantauan adalah bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja solusi Amazon S3 Anda sehingga Anda dapat lebih mudah men-debug kegagalan akses. Logging dapat memberikan wawasan tentang kesalahan yang diterima pengguna, dan kapan dan permintaan apa yang dibuat. AWS menyediakan beberapa alat untuk memantau sumber daya Amazon S3 Anda, seperti berikut ini:

- AWS CloudTrail
- Log Akses Amazon S3
- AWS Trusted Advisor
- Amazon CloudWatch

Untuk informasi selengkapnya, lihat Pencatatan log dan pemantauan di Amazon S3.

Identity and Access Management untuk Amazon S3

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diotorisasi (memiliki izin) untuk menggunakan sumber daya Amazon S3. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Topik

- Audiens
- Mengautentikasi dengan identitas
- Mengelola akses menggunakan kebijakan
- Bagaimana Amazon S3 bekerja dengan IAM
- Kebijakan dan izin di Amazon S3
- Kebijakan bucket untuk Amazon S3
- Kebijakan berbasis identitas untuk Amazon S3

- Panduan yang menggunakan kebijakan untuk mengelola akses ke sumber daya Amazon S3
- Bagaimana Amazon S3 Mengotorisasi Permintaan
- AWS kebijakan terkelola untuk Amazon S3
- Menggunakan Peran Terkait Layanan untuk Lensa Penyimpanan Amazon S3
- Memecahkan masalah identitas dan akses Amazon S3

Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan di Amazon S3.

Pengguna layanan - Jika Anda menggunakan layanan Amazon S3 untuk melakukan pekerjaan Anda, administrator Anda memberi Anda kredensyal dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak fitur Amazon S3 untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fitur di Amazon S3, lihat. Memecahkan masalah identitas dan akses Amazon S3

Administrator layanan - Jika Anda bertanggung jawab atas sumber daya Amazon S3 di perusahaan Anda, Anda mungkin memiliki akses penuh ke Amazon S3. Tugas Anda adalah menentukan fitur dan sumber daya Amazon S3 mana yang harus diakses pengguna layanan Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep Basic IAM. Untuk mempelajari lebih lanjut tentang bagaimana perusahaan Anda dapat menggunakan IAM dengan Amazon S3, lihat. Bagaimana Amazon S3 bekerja dengan IAM

Administrator IAM - Jika Anda administrator IAM, Anda mungkin ingin mempelajari detail tentang cara menulis kebijakan untuk mengelola akses ke Amazon S3. Untuk melihat contoh kebijakan berbasis identitas Amazon S3 yang dapat Anda gunakan di IAM, lihat. Kebijakan berbasis identitas untuk Amazon S3

Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensi identitas Anda. Anda harus diautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengasumsikan peran IAM.

Anda dapat masuk AWS sebagai identitas federasi dengan menggunakan kredensil yang disediakan melalui sumber identitas. AWS IAM Identity Center Pengguna (IAM Identity Center), autentikasi masuk tunggal perusahaan Anda, dan kredensyal Google atau Facebook Anda adalah contoh identitas federasi. Saat Anda masuk sebagai identitas terfederasi, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal AWS akses. Untuk informasi selengkapnya tentang masuk AWS, lihat Cara masuk ke Panduan AWS Sign-In Pengguna Anda Akun AWS.

Jika Anda mengakses AWS secara terprogram, AWS sediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensyal Anda. Jika Anda tidak menggunakan AWS alat, Anda harus menandatangani permintaan sendiri. Untuk informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, lihat Menandatangani permintaan AWS API di Panduan Pengguna IAM.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, lihat Autentikasi multi-faktor dalam Panduan Pengguna AWS IAM Identity Center dan Menggunakan autentikasi multi-faktor (MFA) dalam AWS dalam Panduan Pengguna IAM.

Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut pengguna Akun AWS root dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihat Tugas yang memerlukan kredensial pengguna root dalam Panduan Pengguna IAM.

Identitas gabungan

Sebagai praktik terbaik, mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensyal sementara.

Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, direktori Pusat Identitas AWS Directory Service, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensyal yang disediakan melalui sumber identitas. Ketika identitas federasi mengakses Akun AWS, mereka mengambil peran, dan peran memberikan kredensi sementara.

Untuk manajemen akses terpusat, kami sarankan Anda menggunakan AWS IAM Identity Center. Anda dapat membuat pengguna dan grup di Pusat Identitas IAM, atau Anda dapat menghubungkan dan menyinkronkan ke sekumpulan pengguna dan grup di sumber identitas Anda sendiri untuk digunakan di semua aplikasi Akun AWS dan aplikasi Anda. Untuk informasi tentang Pusat Identitas IAM, lihat Apakah itu Pusat Identitas IAM? dalam Panduan Pengguna AWS IAM Identity Center .

Pengguna dan grup IAM

Pengguna IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, kami merekomendasikan untuk mengandalkan kredensial sementara, bukan membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan tertentu yang memerlukan kredensial jangka panjang dengan pengguna IAM, kami merekomendasikan Anda merotasi kunci akses. Untuk informasi selengkapnya, lihat Merotasi kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensial jangka panjang dalam Panduan Pengguna IAM.

Grup IAM adalah identitas yang menentukan sekumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin bagi beberapa pengguna sekaligus. Grup mempermudah manajemen izin untuk sejumlah besar pengguna sekaligus. Misalnya, Anda dapat memiliki grup yang bernama IAMAdmins dan memberikan izin ke grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari selengkapnya, lihat Kapan harus membuat pengguna IAM (bukan peran) dalam Panduan Pengguna IAM.

Peran IAM

Peran IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Anda dapat mengambil peran IAM untuk sementara AWS Management Console dengan beralih peran. Anda dapat mengambil peran

dengan memanggil operasi AWS CLI atau AWS API atau dengan menggunakan URL kustom. Untuk informasi selengkapnya tentang cara menggunakan peran, lihat Menggunakan peran IAM dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna terfederasi – Untuk menetapkan izin ke identitas terfederasi, Anda membuat peran dan menentukan izin untuk peran tersebut. Ketika identitas terfederasi mengautentikasi, identitas tersebut terhubung dengan peran dan diberi izin yang ditentukan oleh peran. Untuk informasi tentang peran untuk federasi, lihat Membuat peran untuk Penyedia Identitas pihak ketiga dalam Panduan Pengguna IAM. Jika menggunakan Pusat Identitas IAM, Anda harus mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM akan mengorelasikan set izin ke peran dalam IAM. Untuk informasi tentang set izin, lihat Set izin dalam Panduan Pengguna AWS IAM Identity Center .
- Izin pengguna IAM sementara – Pengguna atau peran IAM dapat mengambil peran IAM guna mendapatkan berbagai izin secara sementara untuk tugas tertentu.
- Akses lintas akun – Anda dapat menggunakan peran IAM untuk mengizinkan seseorang (prinsipal tepercaya) di akun lain untuk mengakses sumber daya di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, dengan beberapa Layanan AWS, Anda dapat melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Untuk mempelajari perbedaan antara peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM.
- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Sebagai contoh, ketika Anda memanggil suatu layanan, biasanya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Sebuah layanan mungkin melakukannya menggunakan izin prinsipal yang memanggil, menggunakan peran layanan, atau peran terkait layanan.
  - Sesi akses teruskan (FAS) — Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.

- Peran layanan – Peran layanan adalah peran IAM yang dijalankan oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Membuat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.
- Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

- Aplikasi yang berjalan di Amazon EC2 — Anda dapat menggunakan peran IAM untuk mengelola kredensyal sementara untuk aplikasi yang berjalan pada instans EC2 dan membuat atau permintaan API. AWS CLI AWS Cara ini lebih dianjurkan daripada menyimpan kunci akses dalam instans EC2. Untuk menetapkan AWS peran ke instans EC2 dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instance yang dilampirkan ke instance. Profil instans berisi peran dan memungkinkan program yang berjalan di instans EC2 mendapatkan kredensial sementara. Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan dalam instans Amazon EC2 dalam Panduan Pengguna IAM.

Untuk mempelajari apakah kita harus menggunakan peran IAM atau pengguna IAM, lihat Kapan harus membuat peran IAM (bukan pengguna) dalam Panduan Pengguna IAM.

Mengelola akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal (pengguna, pengguna root, atau sesi peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang struktur dan isi dokumen kebijakan JSON, lihat Gambaran umum kebijakan JSON dalam Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat

kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk melakukan operasinya. Misalnya, anggaplah Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut bisa mendapatkan informasi peran dari AWS Management Console, API AWS CLI, atau AWS API.

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan yang dikelola. Kebijakan inline disematkan langsung ke satu pengguna, grup, atau peran. Kebijakan terkelola adalah kebijakan mandiri yang dapat dilampirkan ke beberapa pengguna, grup, dan peran dalam. Akun AWS Kebijakan AWS terkelola mencakup kebijakan terkelola dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan yang dikelola atau kebijakan inline, lihat Memilih antara kebijakan yang dikelola dan kebijakan inline dalam Panduan Pengguna IAM.

Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan AWS terkelola dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACL)

Daftar kontrol akses (ACL) mengendalikan prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun kebijakan tersebut tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan VPC adalah contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, lihat Gambaran umum daftar kontrol akses (ACL) dalam Panduan Developer Amazon Simple Storage Service.

Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Jenis-jenis kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda oleh jenis kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan tempat Anda mengatur izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas ke entitas IAM (pengguna IAM atau peran IAM). Anda dapat menetapkan batasan izin untuk suatu entitas. Izin yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas milik entitas dan batasan izinnya. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran dalam bidang `Principal` tidak dibatasi oleh batasan izin. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya tentang batasan izin, lihat Batasan izin untuk entitas IAM dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCP) — SCP adalah kebijakan JSON yang menentukan izin maksimum untuk organisasi atau unit organisasi (OU) di. AWS Organizations AWS Organizations adalah layanan untuk mengelompokkan dan mengelola secara terpusat beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur di organisasi, Anda dapat menerapkan kebijakan kontrol layanan (SCP) ke salah satu atau semua akun Anda. SCP membatasi izin untuk entitas di akun anggota, termasuk masing-masing. Pengguna root akun AWS Untuk informasi selengkapnya tentang Organisasi dan SCP, lihat Cara kerja SCP dalam Panduan Pengguna AWS Organizations .
- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter ketika Anda membuat sesi sementara secara programatis untuk peran atau pengguna terfederasi. Izin sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya, lihat Kebijakan sesi dalam Panduan Pengguna IAM.

Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat Logika evaluasi kebijakan di Panduan Pengguna IAM.

Bagaimana Amazon S3 bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke Amazon S3, pelajari fitur IAM apa yang tersedia untuk digunakan dengan Amazon S3.

Fitur IAM yang dapat Anda gunakan dengan Amazon S3

Fitur IAM	Dukungan Amazon S3
Kebijakan berbasis identitas	Ya
Kebijakan berbasis sumber daya	Ya
Tindakan kebijakan	Ya
Sumber daya kebijakan	Ya
kunci-kunci persyaratan kebijakan (spesifik layanan)	Ya
ACL	Ya
ABAC (tanda dalam kebijakan)	Parsial
Kredensial sementara	Ya
Sesi akses teruskan (FAS)	Ya
Peran layanan	Ya
Peran terkait layanan	Parsial

Untuk mendapatkan tampilan tingkat tinggi tentang cara kerja Amazon S3 dan layanan AWS lainnya dengan sebagian besar fitur IAM, AWS lihat layanan yang bekerja dengan IAM di Panduan Pengguna IAM.

Kebijakan berbasis identitas untuk Amazon S3

Mendukung kebijakan berbasis identitas	Ya

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan secara spesifik apakah tindakan dan sumber daya diizinkan atau ditolak, serta kondisi yang menjadi dasar dikabulkan atau ditolaknya tindakan tersebut. Anda tidak dapat menentukan secara spesifik prinsipal dalam sebuah kebijakan berbasis identitas karena prinsipal berlaku bagi pengguna atau peran yang melekat kepadanya. Untuk mempelajari semua elemen yang dapat Anda gunakan dalam kebijakan JSON, lihat Referensi elemen kebijakan JSON IAM dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk Amazon S3

Untuk melihat contoh kebijakan berbasis identitas Amazon S3, lihat. Kebijakan berbasis identitas untuk Amazon S3

Kebijakan berbasis sumber daya dalam Amazon S3

Mendukung kebijakan berbasis sumber daya	Ya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus

menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan secara spesifik seluruh akun atau entitas IAM di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya. Menambahkan prinsipal akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika prinsipal dan sumber daya berbeda Akun AWS, administrator IAM di akun tepercaya juga harus memberikan izin entitas utama (pengguna atau peran) untuk mengakses sumber daya. Mereka memberikan izin dengan melampirkan kebijakan berbasis identitas kepada entitas. Namun, jika kebijakan berbasis sumber daya memberikan akses ke prinsipal dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk informasi selengkapnya, lihat Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM.

Layanan Amazon S3 mendukung kebijakan bucket, kebijakan titik akses, dan hibah akses:

- Kebijakan bucket adalah kebijakan berbasis sumber daya yang dilampirkan ke bucket Amazon S3. Kebijakan bucket menentukan prinsipal mana yang dapat melakukan tindakan di bucket.
- Kebijakan titik akses adalah kebijakan berbasis sumber daya yang dievaluasi bersama dengan kebijakan bucket yang mendasarinya.
- Hibah akses adalah model yang disederhanakan untuk menentukan izin akses ke data di Amazon S3 berdasarkan awalan, bucket, atau objek. Untuk informasi tentang Hibah Akses S3, lihat. Mengelola akses dengan S3 Access Grants

Prinsipal untuk kebijakan bucket

`Principal`Elemen menentukan pengguna, akun, layanan, atau entitas lain yang diizinkan atau ditolak akses ke sumber daya. Berikut ini adalah contoh-contoh menentukan `Principal`. Untuk informasi selengkapnya, lihat Pengguna Utama dalam Panduan Pengguna IAM.

Berikan izin ke Akun AWS

Untuk memberikan izin ke Akun AWS, identifikasi akun menggunakan format berikut.

```
"AWS":"account-ARN"
```

Berikut ini adalah beberapa contohnya.

```
"Principal":{"AWS":"arn:aws:iam::AccountIDWithoutHyphens:root"}
```

```
"Principal":{"AWS":
["arn:aws:iam::AccountID1WithoutHyphens:root","arn:aws:iam::AccountID2WithoutHyphens:root"]}
```

Berikan Izin kepada Pengguna IAM

Untuk memberikan izin kepada pengguna IAM dalam akun Anda, Anda harus memberikan pasangan nama-nilai `"AWS":"user-ARN"`.

```
"Principal":{"AWS":"arn:aws:iam::account-number-without-hyphens:user/username"}
```

Untuk contoh rinci yang memberikan step-by-step instruksi, lihat Contoh 1: Pemilik bucket yang memberikan izin bucket kepada penggunanya danContoh 3: Pemilik bucket yang memberikan izin kepada penggunanya ke objek yang bukan miliknya.

> **Note**
>
> Jika identitas IAM dihapus setelah Anda memperbarui kebijakan bucket, kebijakan bucket akan menampilkan pengenal unik di elemen utama, bukan ARN. ID unik ini tidak pernah digunakan kembali, sehingga Anda dapat menghapus pengguna utama dengan aman dengan pengenal unik dari semua pernyataan kebijakan Anda. Untuk informasi selengkapnya tentang pengidentifikasi unik, lihat pengidentifikasi IAM di Panduan Pengguna IAM.

Memberikan izin anonim

> **Warning**
>
> Berhati-hatilah saat memberikan akses anonim ke bucket Amazon S3. Saat Anda memberikan akses anonim, siapa pun di dunia dapat mengakses bucket Anda. Kami sangat menyarankan agar Anda tidak pernah memberikan akses menulis anonim apa pun ke bucket S3 Anda.

Untuk memberikan izin kepada semua orang, juga disebut sebagai akses anonim, Anda atur wildcard (`"*"`) sebagai nilai `Principal`. Misalnya, jika Anda mengonfigurasi bucket sebagai situs web, maka Anda ingin semua objek yang ada dalam bucket dapat diakses publik.

```
"Principal":"*"
```

```
"Principal":{"AWS":"*"}
```

Menggunakan `"Principal": "*"` dengan `Allow` efek dalam kebijakan berbasis sumber daya memungkinkan siapa pun, meskipun mereka tidak masuk AWS, untuk mengakses sumber daya Anda.

Menggunakan `"Principal" : { "AWS" : "*" }` dengan efek `Allow` dalam kebijakan berbasis sumber daya memungkinkan setiap pengguna root, pengguna IAM, sesi peran yang dianggap, atau pengguna gabungan di akun apa pun di dalam partisi yang sama untuk mengakses sumber daya Anda.

Untuk pengguna anonim, kedua metode ini setara. Untuk informasi selengkapnya, lihat Semua pengguna utama di Panduan Pengguna IAM.

Anda tidak dapat menggunakan wildcard untuk mencocokkan sebagian nama pengguna utama atau ARN.

> **Important**
>
> Karena siapa pun dapat membuat Akun AWS, tingkat keamanan kedua metode ini setara, meskipun fungsinya berbeda.

Batasi izin sumber daya

Anda juga dapat menggunakan kebijakan sumber daya untuk membatasi akses ke sumber daya yang seharusnya tersedia untuk pengguna utama IAM. Gunakan `Deny` pernyataan untuk mencegah akses.

Contoh berikut memblokir akses jika protokol transport aman tidak digunakan:

```
{"Effect": "Deny",
 "Principal": "*",
 "Action": "s3:*",
 "Resource": <bucket ARN>,
 "Condition": {
   "Boolean": { "aws:SecureTransport" : "false"}
 }
}
```

Menggunakan `"Principal": "*"` sehingga pembatasan ini berlaku untuk semua orang adalah praktik terbaik untuk kebijakan ini, daripada mencoba menolak akses hanya ke akun atau pengguna utama tertentu yang menggunakan metode ini.

Memerlukan akses melalui CloudFront URL

Anda dapat meminta pengguna mengakses konten Amazon S3 Anda hanya dengan menggunakan URL, bukan CloudFront URL Amazon S3. Untuk melakukan ini, buat kontrol akses CloudFront asal (OAC). Kemudian, ubah izin pada data S3 Anda. Dalam kebijakan bucket, Anda dapat menetapkan CloudFront Principal sebagai berikut:

```
"Principal":{"Service":"cloudfront.amazonaws.com"}
```

Gunakan `Condition` elemen dalam kebijakan CloudFront untuk mengizinkan akses bucket hanya jika permintaan tersebut atas nama CloudFront distribusi yang berisi asal S3.

```
        "Condition": {
           "StringEquals": {
              "AWS:SourceArn":
 "arn:aws:cloudfront::111122223333:distribution/CloudFront-distribution-ID"
           }
        }
```

Untuk informasi selengkapnya tentang mewajibkan akses S3 melalui CloudFront URL, lihat Membatasi akses ke asal Layanan Penyimpanan Sederhana Amazon di Panduan Pengembang Amazon CloudFront . Untuk informasi selengkapnya tentang manfaat keamanan dan privasi menggunakan Amazon CloudFront, lihat Mengonfigurasi akses aman dan membatasi akses ke konten.

Contoh kebijakan berbasis sumber daya untuk Amazon S3

- Untuk melihat contoh kebijakan untuk bucket Amazon S3, lihat. Kebijakan bucket untuk Amazon S3
- Untuk melihat contoh kebijakan untuk titik akses, lihatMengonfigurasi kebijakan IAM untuk menggunakan titik akses.

Tindakan kebijakan untuk Amazon S3

Mendukung tindakan kebijakan	Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan operasi AWS API terkait. Ada beberapa pengecualian, misalnya tindakan hanya izin yang tidak memiliki operasi API yang cocok. Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam suatu kebijakan. Tindakan tambahan ini disebut tindakan dependen.

Menyertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Berikut ini menunjukkan berbagai jenis hubungan pemetaan antara operasi API S3 dan tindakan kebijakan yang diperlukan.

- O ne-to-one pemetaan dengan nama yang sama. Misalnya, untuk menggunakan operasi `PutBucketPolicy` API, tindakan `s3:PutBucketPolicy` kebijakan diperlukan.
- O ne-to-one pemetaan dengan nama yang berbeda. Misalnya, untuk menggunakan operasi `ListObjectsV2` API, tindakan `s3:ListBucket` kebijakan diperlukan.
- O ne-to-many pemetaan. Misalnya, untuk menggunakan operasi `HeadObject` API, `s3:GetObject` diperlukan. Selain itu, saat Anda menggunakan Kunci Objek S3 dan ingin mendapatkan status Penahanan Hukum atau pengaturan penyimpanan objek, tindakan terkait `s3:GetObjectLegalHold` atau `s3:GetObjectRetention` kebijakan juga diperlukan sebelum Anda dapat menggunakan operasi `HeadObject` API.
- any-to-one Pemetaan M. Misalnya, untuk menggunakan operasi `ListObjectsV2` atau `HeadBucket` API, tindakan `s3:ListBucket` kebijakan diperlukan.

Untuk melihat daftar tindakan Amazon S3 untuk digunakan dalam kebijakan, lihat Tindakan yang ditentukan oleh Amazon S3 di Referensi Otorisasi Layanan. Untuk daftar lengkap operasi Amazon S3 API, lihat Tindakan API Amazon S3 di Referensi API Layanan Penyimpanan Sederhana Amazon.

Tindakan kebijakan di Amazon S3 menggunakan awalan berikut sebelum tindakan:

```
s3
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan tersebut dengan koma.

```
"Action": [
      "s3:action1",
      "s3:action2"
         ]
```

Operasi bucket

Operasi bucket adalah operasi API S3 yang beroperasi pada tipe sumber daya bucket. Misalnya,`CreateBucket`,`ListObjectsV2`, dan`PutBucketPolicy`. Tindakan kebijakan S3 untuk operasi bucket memerlukan `Resource` elemen dalam kebijakan bucket atau kebijakan berbasis identitas IAM sebagai pengidentifikasi Amazon Resource Name (ARN) tipe bucket S3 dalam format contoh berikut.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
```

Kebijakan bucket berikut memberi pengguna *Akua* dengan akun *12345678901* `s3:ListBucket` izin untuk melakukan operasi API ListObjectsV2 dan mencantumkan objek dalam bucket S3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow Akua to list objects in the bucket",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::12345678901:user/Akua"
            },
            "Action": [
            "s3:ListBucket"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
        }
    ]
```

```
}
```

Operasi bucket dalam kebijakan titik akses

Izin yang diberikan dalam kebijakan jalur akses hanya efektif jika bucket yang mendasarinya mengizinkan izin yang sama. Saat Anda menggunakan Titik Akses S3, Anda harus mendelegasikan kontrol akses dari bucket ke titik akses atau menambahkan izin yang sama dalam kebijakan titik akses ke kebijakan bucket yang mendasarinya. Untuk informasi selengkapnya, lihat Mengonfigurasi kebijakan IAM untuk menggunakan titik akses. Dalam kebijakan titik akses, tindakan kebijakan S3 untuk operasi bucket mengharuskan Anda menggunakan `accesspoint` ARN untuk elemen `Resource` dalam format berikut.

```
"Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/DOC-EXAMPLE-ACCESS-POINT"
```

Kebijakan titik akses berikut memberi pengguna *Akua* dengan akun *12345678901* `s3:ListBucket` izin untuk melakukan operasi ListObjectsV2 API melalui titik akses S3 *DOC-EXAMPLE-ACCESS-POINT* untuk mencantumkan objek di bucket terkait titik akses.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow Akua to list objects in the bucket through access point",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::12345678901:user/Akua"
            },
            "Action": [
            "s3:ListBucket"
            ],
            "Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/DOC-EXAMPLE-
ACCESS-POINT"
        }
    ]
}
```

> **Note**
>
> Tidak semua operasi bucket didukung oleh S3 Access Point. Untuk informasi selengkapnya, lihat Kompatibilitas titik akses dengan operasi S3.

Operasi objek

Operasi objek adalah operasi API S3 yang bertindak berdasarkan jenis sumber daya objek. Misalnya,`GetObject`,`PutObject`, dan`DeleteObject`. Tindakan kebijakan S3 untuk operasi objek memerlukan `Resource` elemen dalam kebijakan menjadi objek ARN S3 dalam format contoh berikut.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
```

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix/*"
```

> **Note**
>
> Objek ARN harus berisi garis miring ke depan setelah nama bucket, seperti yang terlihat pada contoh sebelumnya.

Kebijakan bucket berikut memberi pengguna *Akua* akun *12345678901* `s3:PutObject` izin untuk melakukan operasi PutObjectAPI guna mengunggah objek ke bucket S3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow Akua to upload objects",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::12345678901:user/Akua"
            },
            "Action": [
            "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        }
    ]
}
```

Operasi objek dalam kebijakan titik akses

Bila Anda menggunakan S3 Access Points untuk mengontrol akses ke operasi objek, Anda dapat menggunakan kebijakan titik akses. Saat Anda menggunakan kebijakan titik akses, tindakan

kebijakan S3 untuk operasi objek mengharuskan Anda menggunakan `accesspoint` ARN untuk elemen `Resource` dalam format berikut:. `arn:aws:s3:region:account-id:accesspoint/access-point-name/object/resource` Untuk operasi objek yang menggunakan titik akses, Anda harus menyertakan `/object/` nilai setelah seluruh titik akses ARN dalam elemen. `Resource` Berikut ini adalah beberapa contoh.

```
"Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/DOC-EXAMPLE-ACCESS-POINT/object/*"
```

```
"Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/DOC-EXAMPLE-ACCESS-POINT/object/prefix/*"
```

Kebijakan titik akses berikut memberi pengguna *Akua* dengan akun *12345678901* `s3:GetObject` izin untuk melakukan operasi GetObjectAPI melalui titik akses *DOC-EXAMPLE-ACCESS-POINT* pada semua objek di bucket terkait titik akses.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow Akua to get objects through access point",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::12345678901:user/Akua"
            },
            "Action": [
            "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/DOC-EXAMPLE-ACCESS-POINT/object/*"
        }
    ]
}
```

> **Note**
>
> Tidak semua operasi objek didukung oleh S3 Access Point. Untuk informasi selengkapnya, lihat Kompatibilitas titik akses dengan operasi S3.

Operasi titik akses

Operasi titik akses adalah operasi API S3 yang beroperasi pada jenis `accesspoint` sumber daya. Misalnya,`CreateAccessPoint`,`DeleteAccessPoint`, dan`GetAccessPointPolicy`. Tindakan kebijakan S3 untuk operasi titik akses hanya dapat digunakan dalam kebijakan berbasis identitas IAM, bukan dalam kebijakan bucket atau kebijakan jalur akses. Operasi titik akses memerlukan `Resource` elemen untuk menjadi `accesspoint` ARN dalam format contoh berikut.

```
"Resource": "arn:aws:s3:us-west-2:123456789012:accesspoint/DOC-EXAMPLE-ACCESS-POINT"
```

Kebijakan berbasis identitas IAM berikut memberikan s3:GetAccessPointPolicy izin untuk melakukan operasi GetAccessPointPolicyAPI pada titik akses S3 DOC-EXAMPLE-ACCESS-POINT.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Grant permission to retrieve the access point policy of access
 point DOC-EXAMPLE-ACCESS-POINT",
            "Effect": "Allow",
            "Action": [
            "s3:GetAccessPointPolicy"
            ],
            "Resource": "arn:aws:s3:*:123456789012:access point/DOC-EXAMPLE-ACCESS-
POINT"
        }
    ]
}
```

Saat Anda menggunakan Titik Akses, untuk mengontrol akses ke operasi bucket, lihatOperasi bucket dalam kebijakan titik akses; untuk mengontrol akses ke operasi objek, lihatOperasi objek dalam kebijakan titik akses. Untuk informasi selengkapnya tentang cara mengonfigurasi kebijakan titik akses, lihatMengonfigurasi kebijakan IAM untuk menggunakan titik akses.

Operasi Objek Lambda Access Point

Dengan Lambda Objek Amazon S3, Anda dapat menambahkan kode Anda sendiri ke `GET`, `LIST`, dan `HEAD` Amazon S3 dan meminta untuk mengubah dan memproses data saat dikembalikan ke aplikasi. Anda dapat membuat permintaan melalui Object Lambda Access Point, yang berfungsi

sama seperti membuat permintaan melalui titik akses lainnya. Untuk informasi selengkapnya, lihat Mengubah objek dengan S3 Lambda Objek.

Untuk informasi selengkapnya tentang cara mengonfigurasi kebijakan untuk operasi Titik Akses Objek Lambda, lihat. Mengonfigurasi kebijakan IAM untuk Titik Akses Lambda Objek

Operasi Titik Akses Multi-Wilayah

Titik Akses Multi-Region menyediakan titik akhir global yang dapat digunakan aplikasi untuk memenuhi permintaan dari bucket S3 yang terletak di beberapa. Wilayah AWS Anda dapat menggunakan Titik Akses Multi-Wilayah untuk membangun aplikasi Multi-wilayah dengan arsitektur yang sama yang digunakan di satu Wilayah, dan kemudian menjalankan aplikasi tersebut di mana saja di dunia. Untuk informasi selengkapnya, lihat Titik Akses Multi-Wilayah di Amazon S3.

Untuk informasi selengkapnya tentang cara mengonfigurasi kebijakan untuk operasi Titik Akses Multi-Wilayah, lihatContoh kebijakan Multi-Region Access Point.

Operasi pekerjaan Batch

Operasi pekerjaan (Operasi Batch) adalah operasi API S3 yang beroperasi pada jenis sumber daya pekerjaan. Misalnya, `DescribeJob` dan `CreateJob`. Tindakan kebijakan S3 untuk operasi pekerjaan hanya dapat digunakan dalam kebijakan berbasis identitas IAM, bukan dalam kebijakan bucket. Selain itu, operasi pekerjaan memerlukan `Resource` elemen dalam kebijakan berbasis identitas IAM menjadi `job` ARN dalam format contoh berikut.

```
"Resource": "arn:aws:s3:*:123456789012:job/*"
```

Kebijakan berbasis identitas IAM berikut memberikan s3:DescribeJob izin untuk melakukan operasi API pada S3 Batch DescribeJobOperations Job DOC-EXAMPLE-JOB.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow describing the Batch operation job DOC-EXAMPLE-JOB",
            "Effect": "Allow",
            "Action": [
            "s3:DescribeJob"
            ],
            "Resource": "arn:aws:s3:*:123456789012:job/DOC-EXAMPLE-JOB"
```

```
        }
    ]
}
```

Operasi konfigurasi Lensa Penyimpanan S3

Untuk informasi selengkapnya tentang cara mengonfigurasi operasi konfigurasi Lensa Penyimpanan S3, lihatIzin Lensa Penyimpanan Amazon S3.

Operasi akun

Operasi akun adalah operasi API S3 yang beroperasi pada tingkat akun. Misalnya, `GetPublicAccessBlock` (untuk akun). Akun bukanlah jenis sumber daya yang ditentukan oleh Amazon S3. Tindakan kebijakan S3 untuk operasi akun hanya dapat digunakan dalam kebijakan berbasis identitas IAM, bukan dalam kebijakan bucket. Selain itu, operasi akun memerlukan `Resource` elemen dalam kebijakan berbasis identitas IAM. `"*"`

Kebijakan berbasis identitas IAM berikut memberikan `s3:GetAccountPublicAccessBlock` izin untuk melakukan operasi GetPublicAccessBlockAPI tingkat akun dan mengambil pengaturan Blok Akses Publik tingkat akun.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"Allow retrieving the account-level Public Access Block settings",
         "Effect":"Allow",
         "Action":[
            "s3:GetAccountPublicAccessBlock"
         ],
         "Resource":[
            "*"
         ]
       }
    ]
}
```

Contoh kebijakan untuk Amazon S3

- Untuk melihat contoh kebijakan berbasis identitas Amazon S3, lihat. Kebijakan berbasis identitas untuk Amazon S3

- Untuk melihat contoh kebijakan berbasis sumber daya Amazon S3, lihat dan. Kebijakan bucket untuk Amazon S3 Mengonfigurasi kebijakan IAM untuk menggunakan titik akses

Sumber daya kebijakan untuk Amazon S3

Mendukung sumber daya kebijakan	Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek yang menjadi target penerapan tindakan. Pernyataan harus menyertakan elemen `Resource` atau `NotResource`. Praktik terbaiknya, tentukan sumber daya menggunakan Amazon Resource Name (ARN). Anda dapat melakukan ini untuk tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, misalnya operasi pencantuman, gunakan wildcard (*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

```
"Resource": "*"
```

Beberapa tindakan API Amazon S3 mendukung banyak sumber daya. Misalnya, `s3:GetObject` mengakses EXAMPLE-RESOURCE-1 dan EXAMPLE-RESOURCE-2, jadi prinsipal harus memiliki izin untuk mengakses kedua sumber daya. Untuk menentukan beberapa sumber daya dalam satu pernyataan, pisahkan ARN dengan koma.

```
"Resource": [
      "EXAMPLE-RESOURCE-1",
      "EXAMPLE-RESOURCE-2"
```

Sumber daya di Amazon S3 adalah bucket, objek, titik akses, atau pekerjaan. Dalam kebijakan, gunakan Amazon Resource Name (ARN) bucket, objek, access point, atau job untuk mengidentifikasi sumber daya.

Untuk melihat daftar lengkap jenis sumber daya Amazon S3 dan ARNnya, lihat Sumber daya yang ditentukan oleh Amazon S3 di Referensi Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat Tindakan yang ditentukan oleh Amazon S3.

Wildcard untuk ARN sumber daya

Anda dapat menggunakan wildcard sebagai bagian dari sumber daya ARN. Anda dapat menggunakan karakter-karakter wildcard (`*` dan `?`) dalam setiap segmen ARN (bagian-bagian yang dipisahkan dengan titik dua). Tanda bintang (`*`) menunjukkan kombinasi apa pun dari nol karakter atau lebih, dan tanda tanya (`?`) menunjukkan karakter tunggal. Anda dapat menggunakan beberapa karakter `*` atau `?` dalam setiap segmen, tetapi wildcard tidak dapat mencakup segmen.

- ARN berikut ini menggunakan wildcard `*` dalam ID relatif bagian dari ARN untuk mengidentifikasi semua objek di bucket `examplebucket`.

  ```
  arn:aws:s3:::examplebucket/*
  ```

- ARN berikut digunakan `*` untuk menunjukkan semua ember dan objek S3.

  ```
  arn:aws:s3:::*
  ```

- ARN berikut ini menggunakan wildcard, `*` dan `?`, pada bagian `relative-ID`. ARN tersebur mengidentifikasi semua objek dalam bucket seperti `example1bucket`, `example2bucket`, `example3bucket`, dan sebagainya.

  ```
  arn:aws:s3:::example?bucket/*
  ```

Variabel kebijakan untuk ARN sumber daya

Anda dapat menggunakan variabel kebijakan di ARN Amazon S3. Pada waktu evaluasi kebijakan, variabel yang ditentukan sebelumnya ini diganti dengan nilai yang sesuai. Misalkan Anda mengatur bucket Anda sebagai kumpulan folder, satu folder untuk masing-masing pengguna Anda. Nama folder sama dengan nama pengguna. Untuk memberikan pengguna izin ke folder mereka, Anda dapat menentukan variabel kebijakan di ARN sumber daya:

```
arn:aws:s3:::bucket_name/developers/${aws:username}/
```

Saat runtime, ketika kebijakan dievaluasi, variabel `${aws:username}` dalam ARN sumber daya diganti dengan nama pengguna orang yang membuat permintaan.

Contoh kebijakan untuk Amazon S3

- Untuk melihat contoh kebijakan berbasis identitas Amazon S3, lihat. Kebijakan berbasis identitas untuk Amazon S3
- Untuk melihat contoh kebijakan berbasis sumber daya Amazon S3, lihat dan. Kebijakan bucket untuk Amazon S3 Mengonfigurasi kebijakan IAM untuk menggunakan titik akses

Kunci kondisi kebijakan untuk Amazon S3

Mendukung kunci kondisi kebijakan khusus layanan	Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan kondisi yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan operator kondisi, misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa elemen `Condition` dalam sebuah pernyataan, atau beberapa kunci dalam elemen `Condition` tunggal, maka AWS akan mengevaluasinya menggunakan operasi `AND` logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS mengevaluasi kondisi menggunakan `OR` operasi logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Sebagai contoh, Anda dapat memberikan izin kepada pengguna IAM untuk mengakses sumber daya hanya jika izin tersebut mempunyai tag yang sesuai dengan nama pengguna IAM mereka. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM: variabel dan tag dalam Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua kunci kondisi AWS global, lihat kunci konteks kondisi AWS global di Panduan Pengguna IAM.

Setiap kunci kondisi Amazon S3 memetakan ke header permintaan nama yang sama yang diizinkan oleh API tempat kondisinya dapat disetel. Kunci kondisi spesifik Amazon S3 menentukan perilaku

header permintaan nama yang sama. Misalnya, kunci kondisi yang `s3:VersionId` digunakan untuk memberikan izin bersyarat untuk
`s3:GetObjectVersion`
izin menentukan perilaku parameter `versionId` kueri yang Anda tetapkan dalam permintaan GET Object.

Untuk melihat daftar kunci kondisi Amazon S3, lihat Kunci kondisi untuk Amazon S3 di Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan kunci kondisi, lihat Tindakan yang ditentukan oleh Amazon S3.

Contoh: Membatasi unggahan objek ke objek dengan kelas penyimpanan tertentu

Misalkan Akun A, yang diwakili oleh ID akun 123456789012, memiliki sebuah bucket. Administrator Akun A ingin membatasi Dave, pengguna di Akun A, sehingga Dave hanya dapat mengunggah objek ke bucket yang disimpan dengan kelas penyimpanan. `STANDARD_IA` Untuk membatasi unggahan objek ke kelas penyimpanan tertentu, administrator Akun A dapat menggunakan kunci `s3:x-amz-storage-class` kondisi, seperti yang ditunjukkan dalam contoh kebijakan bucket berikut.

```
{
                "Version": "2012-10-17",
                "Statement": [
                  {
                    "Sid": "statement1",
                    "Effect": "Allow",
                    "Principal": {
                      "AWS": "arn:aws:iam::123456789012:user/Dave"
                    },
                    "Action": "s3:PutObject",
                    "Resource": "arn:aws:s3:::example-s3-bucket1/*",
                    "Condition": {
                      "StringEquals": {
                        "s3:x-amz-storage-class": [
                          "STANDARD_IA"
                        ]
                      }
                    }
                  }
                ]
            }
```

Dalam contoh ini, blok `Condition` menentukan syarat `StringEquals` yang diterapkan untuk pasangan nilai kunci tertentu, `"s3:x-amz-acl":["public-read"]`. Ada satu pengaturan kunci yang telah ditetapkan yang dapat Anda gunakan dalam mengekspresikan syarat. Contoh tersebut menggunakan kunci kondisi `s3:x-amz-acl`. Syarat ini mengharuskan pengguna untuk menyertakan header `x-amz-acl` dengan nilai `public-read` dalam setiap permintaan objek PUT.

Contoh kebijakan untuk Amazon S3

- Untuk melihat contoh kebijakan berbasis identitas Amazon S3, lihat. Kebijakan berbasis identitas untuk Amazon S3
- Untuk melihat contoh kebijakan berbasis sumber daya Amazon S3, lihat dan. Kebijakan bucket untuk Amazon S3 Mengonfigurasi kebijakan IAM untuk menggunakan titik akses

ACL di Amazon S3

Mendukung ACL	Ya

Di Amazon S3, kontrol daftar kontrol akses (ACL) yang Akun AWS memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun kebijakan tersebut tidak menggunakan format dokumen kebijakan JSON.

> **Important**
>
> Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL.

Untuk informasi tentang menggunakan ACL untuk mengontrol akses di Amazon S3, lihat. Mengelola Akses dengan ACL

ABAC dengan Amazon S3

Mendukung ABAC (tanda dalam kebijakan)	Parsial

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang menentukan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke entitas IAM (pengguna

atau peran) dan ke banyak AWS sumber daya. Penandaan ke entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian rancanglah kebijakan ABAC untuk mengizinkan operasi ketika tag milik prinsipal cocok dengan tag yang ada di sumber daya yang ingin diakses.

ABAC sangat berguna di lingkungan yang berkembang dengan cepat dan berguna di situasi saat manajemen kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tag, berikan informasi tentang tag di elemen kondisi dari kebijakan menggunakan kunci kondisi `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, atau `aws:TagKeys`.

Jika sebuah layanan mendukung ketiga kunci kondisi untuk setiap jenis sumber daya, nilainya adalah Ya untuk layanan tersebut. Jika suatu layanan mendukung ketiga kunci kondisi untuk hanya beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi selengkapnya tentang ABAC, lihat Apa itu ABAC? dalam Panduan Pengguna IAM. Untuk melihat tutorial yang menguraikan langkah-langkah pengaturan ABAC, lihat Menggunakan kontrol akses berbasis atribut (ABAC) dalam Panduan Pengguna IAM.

Untuk melihat contoh kebijakan berbasis identitas untuk membatasi akses ke pekerjaan Operasi Batch S3 berdasarkan tag, lihat. Mengontrol izin untuk Operasi Batch S3 menggunakan tanda tugas

ABAC dan tag objek

Dalam kebijakan ABAC, objek menggunakan `s3:` tag, bukan `aws:` tag. Untuk mengontrol akses ke objek berdasarkan tag objek, Anda memberikan informasi tag dalam elemen kondisi kebijakan menggunakan tag berikut:

- `s3:ExistingObjectTag/tag-key`
- `s3:s3:RequestObjectTagKeys`
- `s3:RequestObjectTag/tag-key`

Untuk informasi tentang penggunaan tag objek untuk mengontrol akses, termasuk contoh kebijakan izin, lihatKebijakan pemberian tag dan kontrol akses.

Menggunakan kredensyal sementara dengan Amazon S3

Mendukung penggunaan kredensial sementara	Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensyal sementara. Untuk informasi tambahan, termasuk yang Layanan AWS bekerja dengan kredensyal sementara, lihat Layanan AWS yang bekerja dengan IAM di Panduan Pengguna IAM.

Anda menggunakan kredensyal sementara jika Anda masuk AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS menggunakan tautan masuk tunggal (SSO) perusahaan Anda, proses tersebut secara otomatis membuat kredensyal sementara. Anda juga akan secara otomatis membuat kredensial sementara ketika Anda masuk ke konsol sebagai seorang pengguna lalu beralih peran. Untuk informasi selengkapnya tentang peralihan peran, lihat Peralihan peran (konsol) dalam Panduan Pengguna IAM.

Anda dapat membuat kredensyal sementara secara manual menggunakan API AWS CLI atau AWS . Anda kemudian dapat menggunakan kredensil sementara tersebut untuk mengakses. AWS AWS merekomendasikan agar Anda menghasilkan kredensyal sementara secara dinamis alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat Kredensial keamanan sementara di IAM.

Teruskan sesi akses untuk Amazon S3

Mendukung sesi akses maju (FAS)	Ya

Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.

- FAS digunakan oleh Amazon S3 untuk melakukan panggilan AWS KMS untuk mendekripsi objek ketika SSE-KMS digunakan untuk mengenkripsi itu. Untuk informasi selengkapnya, lihat Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS).
- S3 Access Grants juga menggunakan FAS. Setelah Anda membuat hibah akses ke data S3 Anda untuk identitas tertentu, penerima hibah meminta kredensi sementara dari S3 Access Grants. S3 Access Grants memperoleh kredensi sementara untuk pemohon dari AWS STS dan memberikan

kredensi kepada pemohon. Untuk informasi selengkapnya, lihat Minta akses ke data Amazon S3 melalui S3 Access Grants.

Peran layanan untuk Amazon S3

Mendukung peran layanan	Ya

Peran layanan adalah peran IAM yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Membuat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

> **Warning**
>
> Mengubah izin untuk peran layanan dapat merusak fungsionalitas Amazon S3. Edit peran layanan hanya jika Amazon S3 memberikan panduan untuk melakukannya.

Peran terkait layanan untuk Amazon S3

Mendukung peran terkait layanan	Parsial

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Amazon S3 mendukung peran terkait layanan untuk Lensa Penyimpanan Amazon S3. Untuk detail tentang membuat atau mengelola peran terkait layanan Amazon S3, lihat. Menggunakan Peran Terkait Layanan untuk Lensa Penyimpanan Amazon S3

Layanan Amazon S3 sebagai Principal

Nama layanan dalam kebijakan	Fitur S3	Informasi lain
`s3.amazonaws.com`	Replikasi S3	Menyiapkan replikasi langsung
`s3.amazonaws.com`	Pemberitahuan acara S3	Notifikasi Peristiwa Amazon S3
`s3.amazonaws.com`	Inventaris S3	Inventaris Amazon S3
`access-grants.s3.amazonaws.com`	Pemberian Akses S3	Mendaftarkan lokasi
`batchoperations.s3.amazonaws.com`	Operasi Batch S3	Memberikan izin untuk Operasi Batch Amazon S3
`logging.s3.amazonaws.com`	Pencatatan Akses Server S3	Mengaktifkan pencatatan akses server Amazon S3
`storage-lens.s3.amazonaws.com`	Lensa Penyimpanan S3	Melihat metrik Lensa Penyimpanan Amazon S3 menggunakan ekspor data

Kebijakan dan izin di Amazon S3

Halaman ini memberikan gambaran umum tentang kebijakan bucket dan kebijakan pengguna di Amazon S3 dan menjelaskan elemen-elemen dasar sebuah kebijakan. Setiap elemen yang tercantum berkaitan dengan perincian lebih lanjut tentang elemen tersebut dan contoh-contoh cara penggunaannya.

Untuk daftar lengkap tindakan, sumber daya, dan ketentuan Amazon S3, lihat Tindakan, sumber daya, dan kunci kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Secara mendasar, sebuah kebijakan berisi elemen-elemen berikut ini:

- Sumber Daya–Bucket Amazon S3, objek, titik akses, atau tugas yang berlaku untuk kebijakan tersebut. Gunakan Amazon Resource Name (ARN) dari bucket, objek, titik akses, atau job untuk mengidentifikasi sumber daya.

Contoh untuk operasi tingkat bucket:

- `"Resource": "arn:aws:s3:::bucket_name"`.

Contoh operasi tingkat objek:

- `"Resource": "arn:aws:s3:::bucket_name/*"` untuk semua benda di dalam bucket.

- `"Resource": "arn:aws:s3:::bucket_name/prefix/*"` untuk objek di bawah prefiks tertentu di bucket.

Untuk informasi selengkapnya, lihat Sumber daya kebijakan untuk Amazon S3.

- Tindakan–Untuk setiap sumber daya, Amazon S3 mendukung serangkaian operasi. Anda mengidentifikasi operasi sumber daya yang Anda izinkan (atau tolak) dengan menggunakan kata kunci tindakan.

  Misalnya, izin `s3:ListBucket` memungkinkan pengguna untuk menggunakan operasi GET Bucket (Cantumkan Objek) Amazon S3. Untuk informasi selengkapnya tentang penggunaan tindakan Amazon S3, lihat Tindakan kebijakan untuk Amazon S3. Untuk daftar lengkap tindakan Amazon S3, lihat Tindakan.

- Efek–Adalah dampak yang terjadi ketika pengguna meminta tindakan tertentu—ini dapat berupa izinkan atau tolak.

  Jika Anda tidak secara jelas memberikan akses (mengizinkan) ke sumber daya, maka akses akan ditolak secara implisit. Anda juga dapat secara eksplisit menolak akses ke sumber daya. Anda dapat melakukan ini untuk memastikan bahwa pengguna tidak dapat mengakses sumber daya, bahkan jika kebijakan yang berbeda memberikan akses. Untuk informasi lebih lanjut, lihat Elemen Kebijakan IAM JSON: Efek.

- Pengguna Utama–Akun atau pengguna yang diizinkan mengakses tindakan dan sumber daya dalam pernyataan. Dalam kebijakan bucket, pengguna utama adalah pengguna, akun, layanan, atau entitas lain yang merupakan penerima izin ini. Untuk informasi selengkapnya, lihat Prinsipal untuk kebijakan bucket.

- Kondisi–kondisi yang mengatur kapan kebijakan berlaku. Anda dapat menggunakan kunci AWS-wide dan kunci khusus Amazon S3 untuk menentukan kondisi dalam kebijakan akses Amazon S3. Untuk informasi selengkapnya, lihat Contoh kebijakan bucket menggunakan tombol kondisi.

Contoh kebijakan bucket berikut menunjukkan elemen efek, pengguna utama, tindakan, dan sumber daya. Kebijakan ini memungkinkan Akua, pengguna dalam akun *Account-ID*,, `s3:GetObjects3:GetBucketLocation`, dan izin `s3:ListBucket` Amazon S3 di bucket. `awsexamplebucket1`

```
{
    "Version": "2012-10-17",
    "Id": "ExamplePolicy01",
    "Statement": [
        {
            "Sid": "ExampleStatement01",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/Akua"
            },
            "Action": [
                "s3:GetObject",
                "s3:GetBucketLocation",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::awsexamplebucket1/*",
                "arn:aws:s3:::awsexamplebucket1"
            ]
        }
    ]
}
```

Untuk informasi bahasa kebijakan selengkapnya, lihat Kebijakan dan izin dalam referensi kebijakan IAM dan IAM JSON di Panduan Pengguna IAM.

Delegasi izin

Jika Akun AWS memiliki sumber daya, ia dapat memberikan izin tersebut kepada yang lain. Akun AWS Akun tersebut kemudian dapat mendelegasikan izin-izin tersebut, atau sebagian izin, kepada pengguna dalam akun. Ini disebut sebagai delegasi izin. Tetapi akun yang menerima izin dari akun lain tidak dapat mendelegasikan izin lintas akun ke akun lain. Akun AWS

bucket Amazon S3 dan kepemilikan objek

bucket dan objek adalah sumber daya Amazon S3. Secara bawaan, hanya pemilik sumber daya yang dapat mengakses sumber daya ini. Pemilik sumber daya mengacu pada Akun AWS yang menciptakan sumber daya. Sebagai contoh:

- Akun AWS Yang Anda gunakan untuk membuat bucket dan mengunggah objek memiliki sumber daya tersebut.
- Jika Anda mengunggah objek menggunakan pengguna AWS Identity and Access Management (IAM) atau kredensyal peran, pengguna atau peran Akun AWS tersebut memiliki objek tersebut.
- Pemilik bucket dapat memberikan izin lintas akun kepada orang lain Akun AWS (atau pengguna di akun lain) untuk mengunggah objek. Dalam hal ini, objek Akun AWS yang mengunggah memiliki objek tersebut. Pemilik bucket tidak memiliki izin pada objek yang dimiliki akun lain, dengan pengecualian berikut ini:
  - Pemilik bucket membayar tagihan. Pemilik bucket dapat menolak akses ke objek apa pun, atau menghapus objek apa pun di dalam bucket, tanpa memandang siapa yang memilikinya.
  - Pemilik bucket dapat mengarsipkan objek apa pun atau memulihkan objek yang diarsipkan, tanpa memandang siapa yang memilikinya. Pengarsipan mengacu pada kelas penyimpanan yang digunakan untuk menyimpan objek. Untuk informasi selengkapnya, lihat Mengelola siklus hidup penyimpanan Anda.

Autentikasi kepemilikan dan permintaan

Semua permintaan ke bucket akan diautentikasi atau tidak diautentikasi. Permintaan yang diautentikasi harus menyertakan nilai tanda tangan yang mengautentikasi pengirim permintaan, permintaan yang tidak diautentikasi tidak harus menyertakannya. Untuk informasi lebih lanjut tentang autentikasi permintaan, lihat Membuat permintaan.

Pemilik bucket dapat mengizinkan permintaan yang tidak diauntentikasi. Misalnya, PUT Objectpermintaan yang tidak diautentikasi diizinkan jika bucket memiliki kebijakan bucket publik, atau saat ACL bucket memberikan `WRITE` atau `FULL_CONTROL` mengakses `All Users` grup atau pengguna anonim secara khusus. Untuk informasi lebih lanjut tentang kebijakan bucket publik dan daftar kontrol akses (ACL) publik, lihat Arti “publik”.

Semua permintaan yang tidak diautentikasi dibuat oleh pengguna anonim. Pengguna ini diwakili dalam ACL oleh `65a011a29cdf8ec533ec3d1ccaae921c` ID pengguna kanonik tertentu. Jika objek diunggah ke bucket melalui permintaan yang tidak terotentikasi, pengguna anonim memiliki objek.

ACL objek bawaan memberikan `FULL_CONTROL` kepada pengguna anonim sebagai pemilik objek. Oleh karena itu, Amazon S3 mengizinkan permintaan yang tidak diautentikasi untuk mengambil objek atau memodifikasi ACL-nya.

Untuk mencegah objek diubah oleh pengguna anonim, kami merekomendasikan agar Anda tidak menerapkan kebijakan bucket yang mengizinkan penulisan publik anonim ke bucket Anda atau menggunakan ACL yang mengizinkan pengguna anonim tulis akses ke bucket Anda. Anda dapat menerapkan perilaku yang direkomendasikan ini dengan menggunakan Blokir Akses Publik Amazon S3.

Untuk informasi lebih lanjut tentang pemblokiran akses publik, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda. Untuk informasi selengkapnya tentang ACL, lihat Gambaran umum daftar kontrol akses (ACL).

> **Important**
>
> Kami menyarankan agar Anda tidak menggunakan kredensi pengguna Akun AWS root untuk membuat permintaan yang diautentikasi. Sebagai gantinya, buat peran IAM dan berikan akses penuh kepada peran tersebut. Kami merujuk ke pengguna dengan peran ini sebagai pengguna administrator. Anda dapat menggunakan kredensyal yang ditetapkan ke peran administrator, bukan kredensyal pengguna Akun AWS root, untuk berinteraksi AWS dan melakukan tugas, seperti membuat bucket, membuat pengguna, dan memberikan izin. Untuk informasi selengkapnya, lihat kredensyal AWS keamanan di Panduan Pengguna IAM dan praktik terbaik Keamanan di IAM di Panduan Pengguna IAM.

Kebijakan bucket untuk Amazon S3

Kebijakan bucket adalah kebijakan berbasis sumber daya yang dapat Anda gunakan untuk memberikan izin akses ke bucket Amazon S3 dan objek di dalamnya. Hanya pemilik bucket yang dapat mengaitkan kebijakan dengan bucket. Izin yang dipasang pada bucket berlaku untuk semua objek di bucket yang dimiliki oleh pemilik bucket. Izin ini tidak berlaku untuk objek yang dimiliki oleh orang lain Akun AWS.

Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda dan untuk menonaktifkan atau

mengaktifkan daftar kontrol akses (ACL). Secara default, Kepemilikan Objek disetel ke pengaturan diberlakukan pemilik Bucket dan semua ACL dinonaktifkan. Pemilik bucket memiliki semua objek di bucket dan mengelola akses ke data secara eksklusif menggunakan kebijakan.

Kebijakan bucket menggunakan bahasa kebijakan berbasis JSON AWS Identity and Access Management (IAM). Anda dapat menggunakan kebijakan bucket untuk menambah atau menolak izin objek dalam bucket. Kebijakan bucket dapat mengizinkan atau menolak permintaan berdasarkan elemen dalam kebijakan. Elemen-elemen ini termasuk pemohon, tindakan S3, sumber daya, dan aspek atau kondisi permintaan (seperti alamat IP yang digunakan untuk membuat permintaan).

Misalnya, Anda dapat membuat kebijakan bucket yang melakukan langkah berikut:

- Berikan izin lintas akun lainnya untuk mengunggah objek ke bucket S3 Anda
- Memastikan bahwa Anda, pemilik bucket, memiliki kendali penuh atas objek yang diunggah

Untuk informasi selengkapnya, lihat Contoh kebijakan bucket Amazon S3.

> **Important**
>
> Anda tidak dapat menggunakan kebijakan bucket untuk mencegah penghapusan atau transisi oleh aturan Siklus Hidup S3. Misalnya, meskipun kebijakan bucket Anda menolak semua tindakan untuk semua prinsipal, konfigurasi Siklus Hidup S3 Anda tetap berfungsi seperti biasa.

Topik di bagian ini memberikan contoh dan menunjukkan cara menambahkan kebijakan bucket di konsol S3. Untuk informasi tentang kebijakan berbasis identitas, lihat. Kebijakan berbasis identitas untuk Amazon S3 Untuk informasi tentang bahasa kebijakan bucket, lihat Kebijakan dan izin di Amazon S3.

Topik

- Menambahkan kebijakan bucket dengan menggunakan konsol Amazon S3
- Megendalikan akses dari titik akhir VPC dengan kebijakan bucket
- Contoh kebijakan bucket Amazon S3
- Contoh kebijakan bucket menggunakan tombol kondisi

Menambahkan kebijakan bucket dengan menggunakan konsol Amazon S3

Anda dapat menggunakan Pembuat Kebijakan AWS dan konsol Amazon S3 untuk menambahkan kebijakan bucket baru, atau mengedit kebijakan bucket yang sudah ada. Kebijakan bucket adalah kebijakan berbasis sumber daya AWS Identity and Access Management (IAM). Anda menambahkan kebijakan bucket ke bucket untuk memberikan izin akses kepada pengguna lain Akun AWS atau IAM untuk bucket dan objek di dalamnya. Izin objek hanya berlaku untuk objek yang dibuat oleh pemilik bucket. Untuk informasi lebih lanjut tentang kebijakan bucket, lihat Identity and Access Management untuk Amazon S3.

Pastikan Anda menyelesaikan peringatan keamanan, kesalahan, peringatan umum, dan saran dari AWS Identity and Access Management Access Analyzer sebelum Anda menyimpan kebijakan Anda. Penganalisis Akses IAM menjalankan pemeriksaan kebijakan untuk memvalidasi kebijakan Anda terhadap tata bahasa kebijakan IAM dan praktik terbaik. Pemeriksaan ini menghasilkan temuan dan memberikan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang fungsional dan sesuai dengan praktik terbaik keamanan. Untuk mempelajari validasi kebijakan menggunakan Penganalisis Akses IAM lebih lanjut, lihat Memvalidasi kebijakan Penganalisis Akses IAM di Panduan Pengguna IAM. Untuk melihat daftar peringatan, kesalahan, dan saran yang ditampilkan oleh Penganalisis Akses IAM, lihat referensi pemeriksaan kebijakan Penganalisis Akses IAM.

Untuk panduan tentang pemecahan masalah kesalahan dengan kebijakan, lihat. Pecahkan masalah kesalahan Akses Ditolak (403 Forbidden) di Amazon S3

Untuk membuat atau mengedit kebijakan bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di daftar Bucket, pilih nama bucket yang ingin Anda buatkan kebijakan bucket atau yang kebijakan bucketnya ingin Anda edit.
4. Pilih tab Izin.
5. Di Bawah Kebijakan bucket, pilih Edit. Halaman Edit kebijakan bucket akan muncul.
6. Di halaman Edit kebijakan bucket, lakukan salah satu hal berikut:
   - Untuk melihat contoh kebijakan bucket di Panduan Pengguna Amazon S3, pilih Contoh kebijakan.

- Untuk membuat kebijakan secara otomatis, atau mengedit JSON di bagian Kebijakan, pilih Pembuat kebijakan.

Jika Anda memilih Policy Generator, AWS Policy Generator akan terbuka di jendela baru.

a. Pada halaman AWS Pembuat Kebijakan, untuk Pilih Jenis Kebijakan, pilih Kebijakan Bucket S3.

b. Tambahkan pernyataan dengan memasukkan informasi di bidang yang disediakan, lalu pilih Tambah Pernyataan. Ulangi langkah ini sebanyak jumlah pernyataan yang ingin Anda tambahkan. Untuk informasi selengkapnya tentang persyaratan kebijakan, lihat Referensi kebijakan IAM JSON di Panduan Pengguna IAM.

> **Note**
>
> Untuk kenyamanan Anda, halaman Edit kebijakan bucket menampilkan Bucket ARN (Nama Sumber Daya Amazon) dari bucket saat ini di atas bidang teks Kebijakan. Anda dapat menyalin ARN ini untuk digunakan dalam pernyataan di halaman Pembuat Kebijakan AWS .

c. Setelah Anda selesai menambahkan pernyataan, pilih Buat Kebijakan.

d. Salin teks kebijakan yang dihasilkan, pilih Tutup, dan kembali ke halaman Edit kebijakan bucket di konsol Amazon S3.

7. Di kotak Kebijakan, edit kebijakan yang ada atau tempel kebijakan bucket dari AWS Policy Generator. Pastikan peringatan keamanan, kesalahan, peringatan umum, dan saran telah ditangani sebelum menyimpan kebijakan.

> **Note**
>
> Kebijakan bucket dibatasi hingga ukuran 20 KB.

8. (Opsional) Pilih Pratinjau akses eksternal di sudut kanan bawah untuk melihat bagaimana kebijakan baru memengaruhi publik dan akses lintas akun ke sumber daya Anda. Sebelum Anda menyimpan kebijakan Anda, Anda dapat memeriksa apakah itu memperkenalkan temuan Penganalisa Akses IAM baru atau menyelesaikan temuan yang ada. Jika Anda tidak melihat penganalisis aktif, pilih Buka Penganalisis Akses untuk membuat penganalisis akun di Penganalisis Akses IAM. Untuk informasi selengkapnya, lihat Pratinjau akses dalam Panduan Pengguna IAM.

9. Pilih Simpan perubahan, yang mengembalikan Anda ke tab Izin.

Megendalikan akses dari titik akhir VPC dengan kebijakan bucket

Anda dapat menggunakan kebijakan bucket Amazon S3 untuk mengontrol akses ke bucket dari titik akhir virtual private cloud (VPC) tertentu atau VPC tertentu. Bagian ini berisi contoh kebijakan bucket yang dapat Anda gunakan untuk mengontrol akses bucket Amazon S3 dari titik akhir VPC. Untuk mempelajari cara mengatur titik akhir VPC, lihat Titik Akhir VPC dalam Panduan Pengguna VPC.

VPC memungkinkan Anda meluncurkan AWS sumber daya ke jaringan virtual yang Anda tentukan. Endpoint VPC memungkinkan Anda membuat koneksi pribadi antara VPC Anda dan VPC lainnya. Layanan AWS Koneksi pribadi ini tidak memerlukan akses melalui internet, melalui koneksi jaringan pribadi virtual (VPN), melalui instance NAT, atau melalui AWS Direct Connect.

Titik akhir VPC untuk Amazon S3 adalah entitas logika dalam VPC yang memungkinkan konektivitas hanya ke Amazon S3. Rute titik akhir VPC mengarahkan permintaan ke Amazon S3 dan mengarahkan respons kembali ke VPC. Titik akhir VPC hanya mengubah bagaimana permintaan diarahkan. Titik akhir publik Amazon S3 dan nama DNS akan terus bekerja dengan titik akhir VPC. Untuk informasi penting tentang penggunaan titik akhir VPC dengan Amazon S3, lihat Titik akhir Gateway dan titik akhir Gateway untuk Amazon S3 di Panduan Pengguna VPC.

Titik akhir VPC untuk Amazon S3 menyediakan dua cara untuk mengontrol akses ke data Amazon S3 Anda:

- Anda dapat mengontrol permintaan, pengguna, atau grup yang diizinkan melalui titik akhir VPC tertentu. Untuk informasi tentang jenis kontrol akses ini, lihat Mengontrol akses ke titik akhir VPC menggunakan kebijakan titik akhir di Panduan Pengguna VPC.
- Anda dapat mengendalikan VPC atau titik akhir VPC memiliki akses ke bucket Anda dengan menggunakan kebijakan bucket Amazon S3. Untuk contoh jenis kontrol akses kebijakan bucket seperti ini, lihat topik-topik yang membahas tentang pembatasan akses.

Topik

- Membatasi akses ke titik akhir VPC kustom
- Membatasi Akses ke VPC Tertentu

> **Important**
>
> Saat menerapkan kebijakan bucket Amazon S3 untuk titik akhir VPC yang dijelaskan di bagian ini, Anda dapat memblokir akses ke bucket secara tidak sengaja. Izin bucket yang dimaksudkan untuk secara kustom membatasi akses bucket ke koneksi yang berasal dari titik akhir VPC Anda dapat memblokir semua koneksi ke bucket tersebut. Untuk informasi tentang cara memperbaiki masalah ini, lihat Bagaimana cara memperbaiki kebijakan bucket saya jika memiliki ID titik akhir VPC atau VPC yang salah? di pusat AWS Support pengetahuan.

Membatasi akses ke titik akhir VPC kustom

Berikut ini adalah contoh kebijakan bucket Amazon S3 yang membatasi akses ke bucket tertentu, `awsexamplebucket1`, hanya dari titik akhir VPC dengan ID `vpce-1a2b3c4d`. Jika titik akhir yang ditentukan tidak digunakan, kebijakan menolak semua akses ke bucket. `aws:SourceVpce`Kondisi menentukan titik akhir. `aws:SourceVpce`Kondisi ini tidak memerlukan Nama Sumber Daya Amazon (ARN) untuk sumber daya titik akhir VPC, hanya ID titik akhir VPC. Untuk informasi lebih lanjut tentang penggunaan kondisi dalam kebijakan, lihat Contoh kebijakan bucket menggunakan tombol kondisi.

> **Important**
>
> - Sebelum menggunakan kebijakan contoh berikut ini, ganti ID titik akhir VPC dengan nilai yang sesuai untuk kasus penggunaan Anda. Jika tidak, Anda tidak akan dapat mengakses bucket Anda.
> - Kebijakan ini menonaktifkan akses konsol ke bucket yang ditentukan karena permintaan konsol tidak berasal dari titik akhir VPC yang ditentukan.

```
{
   "Version": "2012-10-17",
   "Id": "Policy1415115909152",
   "Statement": [
     {
       "Sid": "Access-to-specific-VPCE-only",
       "Principal": "*",
       "Action": "s3:*",
       "Effect": "Deny",
```

```
        "Resource": ["arn:aws:s3:::awsexamplebucket1",
                     "arn:aws:s3:::awsexamplebucket1/*"],
        "Condition": {
          "StringNotEquals": {
            "aws:SourceVpce": "vpce-1a2b3c4d"
          }
        }
      }
    ]
}
```

Membatasi Akses ke VPC Tertentu

Anda dapat membuat kebijakan bucket yang membatasi akses ke VPC tertentu dengan menggunakan kondisi `aws:SourceVpc` tersebut. Hal ini berguna jika Anda memiliki beberapa titik akhir VPC yang dikonfigurasi dalam VPC yang sama, dan Anda ingin mengelola akses ke bucket Amazon S3 Anda untuk semua titik akhir Anda. Berikut ini adalah contoh kebijakan yang menolak akses ke `awsexamplebucket1` dan objeknya dari siapa pun di luar VPC`vpc-111bbb22`. Jika VPC yang ditentukan tidak digunakan, kebijakan menolak semua akses ke bucket. Pernyataan ini tidak memberikan akses ke bucket. Untuk memberikan akses, Anda harus menambahkan `Allow` pernyataan terpisah. Kunci `vpc-111bbb22` kondisi tidak memerlukan ARN untuk sumber daya VPC, hanya ID VPC.

> **Important**
>
> - Sebelum menggunakan kebijakan contoh berikut, ganti ID VPC dengan nilai yang sesuai untuk kasus penggunaan Anda. Jika tidak, Anda tidak akan dapat mengakses bucket Anda.
> - Kebijakan ini menonaktifkan akses konsol ke bucket yang ditentukan karena permintaan konsol tidak berasal dari VPC yang ditentukan.

```
{
   "Version": "2012-10-17",
   "Id": "Policy1415115909153",
   "Statement": [
     {
       "Sid": "Access-to-specific-VPC-only",
       "Principal": "*",
```

```
        "Action": "s3:*",
        "Effect": "Deny",
        "Resource": ["arn:aws:s3:::awsexamplebucket1",
                     "arn:aws:s3:::awsexamplebucket1/*"],
        "Condition": {
          "StringNotEquals": {
            "aws:SourceVpc": "vpc-111bbb22"
          }
        }
      }
    ]
}
```

Contoh kebijakan bucket Amazon S3

Dengan kebijakan bucket Amazon S3, Anda dapat mengamankan akses ke objek di bucket, sehingga hanya pengguna dengan izin yang sesuai yang dapat mengaksesnya. Anda bahkan dapat mencegah pengguna yang diautentikasi tanpa izin yang sesuai untuk mengakses sumber daya Amazon S3 Anda.

Bagian ini menyajikan contoh kasus penggunaan umum untuk kebijakan bucket. Kebijakan sampel ini digunakan *`example-s3-bucket`* sebagai nilai sumber daya. Untuk menguji kebijakan ini, ganti *`user input placeholders`* dengan informasi Anda sendiri (seperti nama bucket Anda).

Untuk memberikan atau menolak izin ke satu set objek, Anda dapat menggunakan karakter wildcard (`*`) di Amazon Resource Names (ARN) dan nilai-nilai lainnya. Misalnya, Anda dapat mengontrol akses ke grup objek yang dimulai dengan awalan umum atau diakhiri dengan ekstensi tertentu, seperti `.html`.

Untuk informasi selengkapnya tentang bahasa kebijakan AWS Identity and Access Management (IAM), lihatKebijakan dan izin di Amazon S3.

> **Note**
>
> Saat menguji izin dengan menggunakan konsol Amazon S3, Anda harus memberikan izin tambahan yang diperlukan konsol—`s3:ListAllMyBuckets`, `s3:GetBucketLocation`, dan. `s3:ListBucket`. Untuk panduan contoh yang memberikan izin kepada pengguna dan menguji izin tersebut dengan menggunakan konsol, lihat. Mengontrol akses ke bucket dengan kebijakan pengguna

Sumber daya tambahan untuk membuat kebijakan bucket meliputi:

- Untuk daftar tindakan kebijakan IAM, sumber daya, dan kunci kondisi yang dapat Anda gunakan saat membuat kebijakan bucket, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.
- Untuk panduan tentang membuat kebijakan S3 Anda, lihat Menambahkan kebijakan bucket dengan menggunakan konsol Amazon S3.
- Untuk memecahkan masalah kesalahan dengan kebijakan, lihat Pecahkan masalah kesalahan Akses Ditolak (403 Forbidden) di Amazon S3.

Topik

- Memberikan izin baca-saja kepada pengguna anonim publik
- Membutuhkan enkripsi
- Mengelola bucket menggunakan ACL terekam
- Mengelola akses objek dengan penandaan objek
- Mengelola akses objek dengan menggunakan kunci kondisi global
- Mengelola akses berdasarkan alamat IP tertentu
- Mengelola akses sesuai permintaan HTTP atau HTTPS
- Mengelola akses pengguna ke folder tertentu
- Mengelola akses log akses
- Mengelola akses ke Amazon CloudFront OAI
- Mengelola akses untuk Lensa Penyimpanan Amazon S3
- Mengelola izin untuk laporan Inventaris S3, S3 analytics, dan Inventaris S3
- Membutuhkan MFA
- Mencegah pengguna menghapus objek

Memberikan izin baca-saja kepada pengguna anonim publik

Anda dapat menggunakan setelan kebijakan untuk memberikan akses ke pengguna anonim publik yang berguna jika Anda mengonfigurasi bucket sebagai situs web statis. Ini mengharuskan Anda untuk menonaktifkan blokir akses publik untuk bucket Anda. Untuk informasi selengkapnya tentang cara melakukannya, dan kebijakan yang diperlukan, lihatMengatur izin untuk akses situs web. Untuk mempelajari cara menyiapkan kebijakan yang lebih ketat untuk tujuan yang sama, lihat Bagaimana

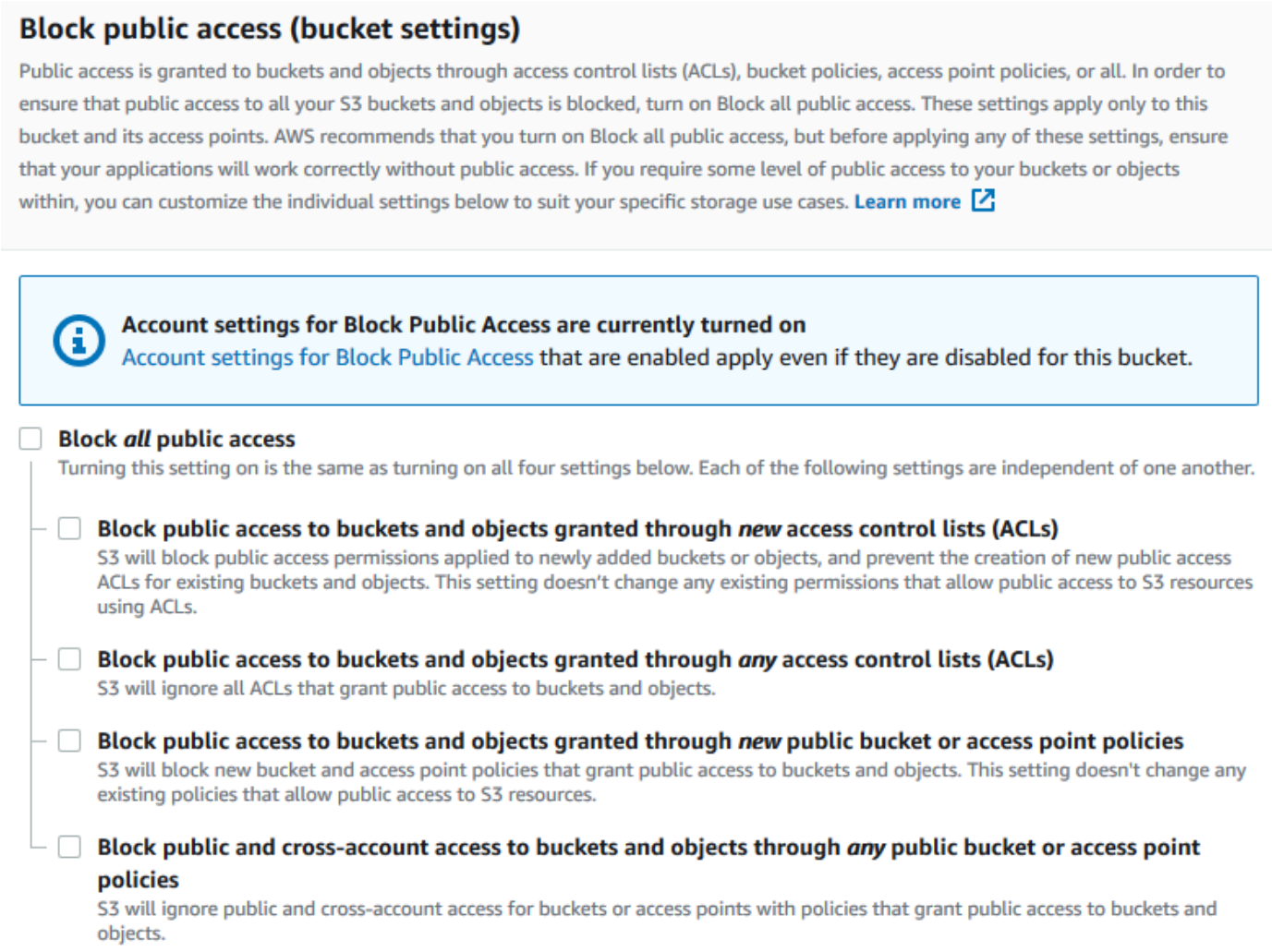
cara memberikan akses baca publik ke beberapa objek di bucket Amazon S3 saya? di pusat AWS pengetahuan.

Secara default, Amazon S3 memblokir akses publik ke akun dan bucket Anda. Jika Anda ingin menggunakan bucket untuk menghosting situs web statis, Anda dapat menggunakan langkah-langkah ini untuk mengedit pengaturan blokir akses publik Anda.

> **Warning**
>
> Sebelum Anda menyelesaikan langkah ini, tinjau Melakukan blok akses publik ke penyimpanan Amazon S3 Anda untuk memastikan bahwa Anda telah memahami dan menerima risiko yang terkait dengan mengizinkan akses publik. Saat Anda mematikan pengaturan blokir akses publik untuk membuat bucket Anda menjadi publik, siapa pun di internet dapat mengakses bucket Anda. Kami sarankan agar Anda memblokir semua akses publik ke bucket Anda.

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pilih nama bucket yang telah Anda konfigurasikan sebagai situs web statis.
3. Pilih Izin.
4. Di bagian bawah Blokir akses publik (pengaturan bucket), pilih Edit.
5. Kosongkan Blokir semua akses publik, lalu pilih Simpan perubahan.

   > **Warning**
   >
   > Sebelum Anda menyelesaikan langkah ini, tinjau Melakukan blok akses publik ke penyimpanan Amazon S3 Anda untuk memastikan bahwa Anda telah memahami dan menerima risiko yang terkait dengan mengizinkan akses publik. Saat Anda mematikan pengaturan blokir akses publik untuk membuat bucket Anda menjadi publik, siapa pun di internet dapat mengakses bucket Anda. Kami sarankan agar Anda memblokir semua akses publik ke bucket Anda.



Amazon S3 menonaktifkan pengaturan Blokir Akses Publik untuk bucket Anda. Untuk membuat situs web publik statis, Anda mungkin harus mengedit pengaturan Blokir Akses Publik untuk akun Anda sebelum menambahkan kebijakan bucket. Jika pengaturan akun untuk Blokir Akses Publik saat ini diaktifkan, Anda akan melihat catatan di Blokir akses publik (pengaturan bucket).

Membutuhkan enkripsi

Membutuhkan SSE-KMS untuk semua objek yang ditulis ke bucket

Contoh kebijakan berikut mengharuskan setiap objek yang ditulis ke bucket dienkripsi dengan enkripsi sisi server menggunakan AWS Key Management Service (AWS KMS) keys (SSE-KMS). Jika objek tidak dienkripsi dengan SSE-KMS, permintaan ditolak.

```
{
"Version": "2012-10-17",
"Id": "PutObjPolicy",
"Statement": [{
  "Sid": "DenyObjectsThatAreNotSSEKMS",
  "Principal": "*",
  "Effect": "Deny",
  "Action": "s3:PutObject",
  "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
  "Condition": {
    "Null": {
      "s3:x-amz-server-side-encryption-aws-kms-key-id": "true"
    }
  }
}]
}
```

Memerlukan SSE-KMS dengan spesifik AWS KMS key untuk semua objek yang ditulis ke bucket

Contoh kebijakan berikut menyangkal objek apa pun ditulis ke bucket jika tidak dienkripsi dengan SSE-KMS dengan menggunakan ID kunci KMS tertentu. Bahkan jika objek dienkripsi dengan SSE-KMS dengan menggunakan header per permintaan atau enkripsi default bucket, objek tidak dapat ditulis ke bucket jika belum dienkripsi dengan kunci KMS yang ditentukan. Pastikan untuk mengganti ARN kunci KMS yang digunakan dalam contoh ini dengan ARN kunci KMS Anda sendiri.

```
{
"Version": "2012-10-17",
"Id": "PutObjPolicy",
"Statement": [{
  "Sid": "DenyObjectsThatAreNotSSEKMSWithSpecificKey",
  "Principal": "*",
  "Effect": "Deny",
  "Action": "s3:PutObject",
  "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
  "Condition": {
    "ArnNotEqualsIfExists": {
      "s3:x-amz-server-side-encryption-aws-kms-key-id": "arn:aws:kms:us-
east-2:111122223333:key/01234567-89ab-cdef-0123-456789abcdef"
    }
  }
}]
}
```

Mengelola bucket menggunakan ACL terekam

memberikan izin ke beberapa akun untuk mengunggah objek atau mengatur ACL objek untuk akses publik

Contoh kebijakan berikut memberikan `s3:PutObjectAcl` izin `s3:PutObject` dan ke beberapa. Akun AWS Selain itu, kebijakan contoh mengharuskan setiap permintaan untuk operasi ini harus menyertakan daftar kontrol akses `public-read` kaleng (ACL). Untuk informasi selengkapnya, lihat Tindakan kebijakan untuk Amazon S3 dan Kunci kondisi kebijakan untuk Amazon S3.

> **Warning**
>
> ACL `public-read` terekam memungkinkan siapa pun di dunia untuk melihat objek di bucket Anda. Berhati-hatilah saat memberikan akses anonim ke bucket Amazon S3 Anda atau menonaktifkan pengaturan blokir akses publik. Saat Anda memberikan akses anonim, siapa pun di dunia dapat mengakses bucket Anda. Kami menyarankan Anda untuk tidak pernah memberikan akses anonim ke bucket Amazon S3 Anda kecuali jika Anda secara kustom memerlukannya, seperti hosting situs web statis. Jika Anda ingin mengaktifkan blokir pengaturan akses publik untuk hosting situs web statis, lihat Tutorial: Mengonfigurasi situs web statis di Amazon S3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AddPublicReadCannedAcl",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root",
                    "arn:aws:iam::444455556666:root"
                ]
            },
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
            "Condition": {
```

```
                "StringEquals": {
                    "s3:x-amz-acl": [
                        "public-read"
                    ]
                }
            }
        }
    ]
}
```

Berikan izin lintas akun untuk unggah objek sekaligus memastikan bahwa pemilik bucket memiliki kendali penuh

Contoh berikut menunjukkan cara mengizinkan orang lain Akun AWS mengunggah objek ke bucket sambil memastikan bahwa Anda memiliki kendali penuh atas objek yang diunggah. Kebijakan ini memberikan kemampuan spesifik Akun AWS (*111122223333*) untuk mengunggah objek hanya jika akun tersebut menyertakan ACL yang `bucket-owner-full-control` dikalengkan saat diunggah. `StringEquals`Kondisi dalam kebijakan tersebut menyebutkan kunci `s3:x-amz-acl` kondisi untuk mengekspresikan persyaratan ACL terekam. Untuk informasi selengkapnya, lihat Kunci kondisi kebijakan untuk Amazon S3.

```
{
   "Version":"2012-10-17",
   "Statement":[
     {
       "Sid":"PolicyForAllowUploadWithACL",
       "Effect":"Allow",
       "Principal":{"AWS":"111122223333"},
       "Action":"s3:PutObject",
       "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
       "Condition": {
         "StringEquals": {"s3:x-amz-acl":"bucket-owner-full-control"}
       }
     }
   ]
}
```

Mengelola akses objek dengan penandaan objek

Mengizinkan pengguna untuk membaca objek yang memiliki kunci dan nilai tag spesifik

Kebijakan izin berikut membatasi pengguna agar hanya membaca objek yang memiliki kunci dan nilai tag `environment: production`. Kebijakan ini menggunakan kunci kondisi `s3:ExistingObjectTag` untuk menentukan kunci tag dan nilai.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Principal":{
            "AWS":"arn:aws:iam::111122223333:role/JohnDoe"
         },
         "Effect":"Allow",
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion"
         ],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
         "Condition":{
            "StringEquals":{
               "s3:ExistingObjectTag/environment":"production"
            }
         }
      }
   ]
}
```

Batasi kunci tag objek mana yang dapat ditambahkan pengguna

Kebijakan contoh berikut memberikan izin pengguna untuk melakukan `s3:PutObjectTagging` tindakan, yang memungkinkan pengguna menambahkan tag ke objek yang sudah ada. Kondisi ini menggunakan kunci kondisi `s3:RequestObjectTagKeys` untuk menentukan kunci tag yang diizinkan, seperti `Owner` atau `CreationDate`. Untuk informasi selengkapnya, lihat Membuat kondisi yang menguji beberapa nilai kunci dalam Panduan Pengguna IAM.

Kebijakan ini memastikan agar setiap kunci tag yang ditentukan dalam permintaan adalah kunci tag yang diotorisasi. Kualifikasi `ForAnyValue` dalam kondisi tersebut memastikan bahwa setidaknya ada satu kunci yang ditentukan dalam permintaan.

```
{
   "Version": "2012-10-17",
  "Statement": [
    {"Principal":{"AWS":[
            "arn:aws:iam::111122223333:role/JohnDoe"
         ]
       },
 "Effect": "Allow",
      "Action": [
        "s3:PutObjectTagging"
      ],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {"ForAnyValue:StringEquals": {"s3:RequestObjectTagKeys": [
            "Owner",
            "CreationDate"
          ]
        }
      }
    }
  ]
}
```

Memerlukan kunci dan nilai tag spesifik saat mengizinkan pengguna untuk menambahkan tag objek

Kebijakan contoh berikut memberikan izin pengguna untuk melakukan `s3:PutObjectTagging` tindakan, yang memungkinkan pengguna menambahkan tag ke objek yang sudah ada. Kondisi ini mengharuskan pengguna untuk menyertakan kunci tag tertentu (seperti *`Project`*) dengan set nilai ke *`X`*.

```
{
   "Version": "2012-10-17",
  "Statement": [
    {"Principal":{"AWS":[
       "arn:aws:iam::111122223333:user/JohnDoe"
         ]
       },
      "Effect": "Allow",
      "Action": [
        "s3:PutObjectTagging"
      ],
```

```
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {"StringEquals": {"s3:RequestObjectTag/Project": "X"
        }
      }
    }
  ]
}
```

Mengizinkan pengguna untuk hanya menambahkan objek dengan kunci dan nilai tag objek tertentu

Kebijakan contoh berikut memberikan izin pengguna untuk melakukan `s3:PutObject` tindakan, sehingga pengguna dapat menambahkan tag ke objek yang sudah ada. Namun, `Condition` pernyataan membatasi kunci tag dan nilai yang diizinkan pada objek yang diunggah. Dalam contoh ini, pengguna hanya dapat menambahkan objek yang memiliki kunci tag tertentu (*`Department`*) dengan nilai yang disetel *`Finance`* ke bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Principal":{
            "AWS":[
                 "arn:aws:iam::111122223333:user/JohnDoe"
         ]
        },
        "Effect": "Allow",
        "Action": [
            "s3:PutObject"
        ],
        "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        ],
        "Condition": {
            "StringEquals": {
                "s3:RequestObjectTag/Department": "Finance"
            }
        }
    }]
}
```

Mengelola akses objek dengan menggunakan kunci kondisi global

Kunci kondisi global adalah kunci konteks kondisi dengan `aws` awalan. Layanan AWS dapat mendukung kunci kondisi global atau kunci khusus layanan yang menyertakan awalan layanan. Anda dapat menggunakan `Condition` elemen kebijakan JSON untuk membandingkan kunci dalam permintaan dengan nilai kunci yang Anda tentukan dalam kebijakan Anda.

Batasi akses hanya ke pengiriman log akses server Amazon S3

Dalam contoh kebijakan bucket berikut, kunci kondisi aws:SourceArnglobal digunakan untuk membandingkan Amazon Resource Name (ARN) sumber daya, membuat service-to-service permintaan dengan ARN yang ditentukan dalam kebijakan. Kunci kondisi `aws:SourceArn` global digunakan untuk mencegah layanan Amazon S3 digunakan sebagai wakil yang bingung selama transaksi antar layanan. Hanya layanan Amazon S3 yang diizinkan untuk menambahkan objek ke bucket Amazon S3.

Contoh kebijakan bucket ini hanya memberikan `s3:PutObject` izin kepada pengguna utama layanan logging (`logging.s3.amazonaws.com`).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowPutObjectS3ServerAccessLogsPolicy",
            "Principal": {
                "Service": "logging.s3.amazonaws.com"
            },
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET-logs/*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111111111111"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:::EXAMPLE-SOURCE-BUCKET"
                }
            }
        },
        {
            "Sid": "RestrictToS3ServerAccessLogs",
            "Effect": "Deny",
```

```
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET-logs/*",
            "Condition": {
                "ForAllValues:StringNotEquals": {
                    "aws:PrincipalServiceNamesList": "logging.s3.amazonaws.com"
                }
            }
        }
    ]
}
```

Izinkan akses hanya ke organisasi Anda

Jika Anda ingin meminta semua kepala sekolah IAM yang mengakses sumber daya berasal dari organisasi Anda (termasuk akun AWS Organizations manajemen), Anda dapat menggunakan kunci kondisi global. Akun AWS `aws:PrincipalOrgID`

Untuk memberikan atau membatasi jenis akses ini, tentukan `aws:PrincipalOrgID` kondisi dan tetapkan nilainya ke ID organisasi Anda dalam kebijakan bucket. ID organisasi digunakan untuk mengontrol akses ke bucket. Saat Anda menggunakan `aws:PrincipalOrgID` kondisi ini, izin dari kebijakan bucket juga diterapkan ke semua akun baru yang ditambahkan ke organisasi.

Berikut adalah contoh kebijakan bucket berbasis sumber daya yang dapat Anda gunakan untuk memberikan pengguna utama IAM tertentu di organisasi Anda akses langsung ke bucket Anda. Dengan menambahkan kunci kondisi `aws:PrincipalOrgID` global ke kebijakan bucket Anda, pengguna utama sekarang harus berada di organisasi Anda untuk mendapatkan akses ke sumber daya. Bahkan jika Anda secara tidak sengaja menentukan akun yang salah saat memberikan akses, kunci kondisi aws:PrincipalOrgID global bertindak sebagai perlindungan tambahan. Ketika kunci global ini digunakan dalam kebijakan, ini mencegah semua pengguna utama dari luar organisasi tertentu mengakses bucket S3. Hanya pengguna utama dari akun di organisasi yang terdaftar yang dapat memperoleh akses ke sumber daya.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "AllowGetObject",
        "Principal": {
            "AWS": "*"
        },
        "Effect": "Allow",
```

```
        "Action": "s3:GetObject",
        "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
        "Condition": {
            "StringEquals": {
                "aws:PrincipalOrgID": ["o-aa111bb222"]
            }
        }
    }]
}
```

Mengelola akses berdasarkan alamat IP tertentu

Membatasi akses ke suatu Wilayah tertentu

Contoh berikut menolak semua pengguna melakukan operasi Amazon S3 pada objek dalam bucket tertentu kecuali permintaan tersebut berasal dari rentang alamat IP yang ditentukan.

> **Note**
>
> Saat membatasi akses ke alamat IP tertentu, pastikan Anda juga menentukan titik akhir VPC, alamat IP sumber VPC, atau alamat IP eksternal yang dapat mengakses bucket S3. Jika tidak, Anda mungkin kehilangan akses ke bucket jika kebijakan Anda menolak semua pengguna melakukan operasi S3 apa pun pada objek di bucket Anda tanpa izin yang tepat.

`Condition`Pernyataan kebijakan ini mengidentifikasi *`192.0.2.0/24`*sebagai rentang alamat IP Protokol Internet versi 4 (IPv4) yang diizinkan.

`Condition`Blok menggunakan `NotIpAddress` kondisi dan kunci `aws:SourceIp` kondisi, yang merupakan kunci kondisi AWS lebar. Kunci kondisi `aws:SourceIp` hanya dapat digunakan untuk rentang alamat IP publik. Untuk informasi lebih lanjut tentang kunci kondisi ini, lihat Kunci kondisi kebijakan untuk Amazon S3. Nilai `aws:SourceIp` IPv4 menggunakan notasi CIDR standar. Untuk informasi lebih lanjut, lihat Referensi Elemen Kebijakan IAM JSON dalam Panduan Pengguna IAM.

> **Warning**
>
> Sebelum menggunakan kebijakan ini, ganti rentang alamat IP *`192.0.2.0/24`* dalam contoh ini dengan nilai yang sesuai untuk kasus penggunaan Anda. Jika tidak, Anda akan kehilangan kemampuan untuk mengakses bucket Anda.

```
{
    "Version": "2012-10-17",
    "Id": "S3PolicyId1",
    "Statement": [
        {
            "Sid": "IPAllow",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": "192.0.2.0/24"
                }
            }
        }
    ]
}
```

Izinkan alamat IPv4 dan IPv6

Saat Anda mulai menggunakan alamat IPv6, kami rekomendasikan agar Anda memperbarui semua kebijakan organisasi dengan rentang alamat IPv6 Anda selain dari rentang alamat IPv4 Anda yang telah ada. Melakukan hal ini akan membantu memastikan bahwa kebijakan terus berfungsi saat Anda melakukan transisi ke IPv6.

Contoh kebijakan bucket berikut menunjukkan cara menggabungkan rentang alamat IPv4 dan IPv6 untuk mencakup semua alamat IP yang valid dalam organisasi Anda. Kebijakan contoh memungkinkan akses ke alamat IP contoh *192.0.2.1*dan *2001:DB8:1234:5678::1*dan menolak akses ke alamat *203.0.113.1*dan *2001:DB8:1234:5678:ABCD::1*.

Kunci kondisi `aws:SourceIp` hanya dapat digunakan untuk rentang alamat IP publik. Nilai IPv6 untuk `aws:SourceIp` harus dalam format standar CIDR. Untuk IPv6, kami support dengan menggunakan `::` untuk mewakili rentang 0s (misalnya, `2001:DB8:1234:5678::/64`). Untuk informasi lebih lanjut, lihat Operator Syarat Alamat IP dalam Panduan Pengguna IAM.

> **Warning**
>
> Ganti rentang alamat IP dalam contoh ini dengan nilai yang sesuai untuk kasus penggunaan Anda sebelum menggunakan kebijakan ini. Jika tidak, Anda mungkin kehilangan kemampuan untuk mengakses bucket Anda.

```
{
    "Id": "PolicyId2",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIPmix",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                },
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "203.0.113.0/24",
                        "2001:DB8:1234:5678:ABCD::/80"
                    ]
                }
            }
        }
    ]
}
```

Mengelola akses sesuai permintaan HTTP atau HTTPS

Batasi Akses Hanya ke Permintaan HTTPS

Jika Anda ingin mencegah penyerang potensial memanipulasi lalu lintas jaringan, Anda dapat menggunakan HTTPS (TLS) untuk hanya mengizinkan koneksi terenkripsi sambil membatasi permintaan HTTP mengakses bucket Anda. Untuk menentukan apakah permintaan tersebut HTTP atau HTTPS, gunakan kunci kondisi aws:SecureTransportglobal dalam kebijakan bucket S3 Anda. Kunci `aws:SecureTransport` kondisi memeriksa apakah permintaan dikirim dengan menggunakan HTTP.

Jika permintaan kembali`true`, maka permintaan dikirim melalui HTTPS. Jika permintaan kembali`false`, maka permintaan dikirim melalui HTTP. Anda kemudian dapat mengizinkan atau menolak akses ke bucket Anda berdasarkan skema permintaan yang diinginkan.

Dalam contoh berikut, kebijakan bucket secara eksplisit menolak permintaan HTTP.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "RestrictToTLSRequestsOnly",
        "Action": "s3:*",
        "Effect": "Deny",
        "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        ],
        "Condition": {
            "Bool": {
                "aws:SecureTransport": "false"
            }
        },
        "Principal": "*"
    }]
}
```

Membatasi akses ke suatu Wilayah tertentu

Misalkan Anda memiliki situs web dengan nama domain *`www.example.com`* atau *`example.com`* dengan tautan ke foto dan video yang disimpan di bucket Anda bernama *`example-s3-bucket`*. Secara default, semua sumber daya Amazon S3 bersifat pribadi, jadi hanya Akun AWS yang membuat sumber daya yang dapat mengaksesnya.

Untuk mengizinkan akses baca ke objek ini dari situs web Anda, Anda dapat menambahkan kebijakan bucket yang mengizinkan `s3:GetObject` izin dengan syarat `GET` permintaan harus berasal dari halaman web tertentu. Kebijakan berikut membatasi permintaan dengan menggunakan `StringLike` kondisi dengan kunci `aws:Referer` kondisi.

```
{
  "Version":"2012-10-17",
  "Id":"HTTP referer policy example",
  "Statement":[
    {
      "Sid":"Allow only GET requests originating from www.example.com and
 example.com.",
      "Effect":"Allow",
      "Principal":"*",
      "Action":["s3:GetObject","s3:GetObjectVersion"],
      "Resource":"arn:aws:s3:::example-s3-bucket/*",
      "Condition":{
        "StringLike":{"aws:Referer":["http://www.example.com/*","http://example.com/
*"]}
      }
    }
  ]
}
```

Pastikan browser yang Anda gunakan menyertakan `referer` header HTTP dalam permintaan.

> **Warning**
>
> Sebaiknya Anda berhati-hati saat menggunakan tombol `aws:Referer` kondisi. Menyertakan nilai header perujuk HTTP yang diketahui publik sangat berbahaya. Pihak yang tidak berwenang dapat menggunakan browser yang diubah atau disesuaikan untuk menyediakan nilai `aws:Referer` yang mereka pilih. Oleh karena itu, jangan gunakan `aws:Referer` untuk mencegah pihak yang tidak berwenang membuat AWS permintaan langsung.
>
> Kunci `aws:Referer` kondisi ditawarkan hanya untuk memungkinkan pelanggan melindungi konten digital mereka, seperti konten yang disimpan di Amazon S3, agar tidak dirujuk di situs pihak ketiga yang tidak sah. Untuk informasi lebih lanjut, lihat aws:Referer dalam Panduan Pengguna IAM.

Mengelola akses pengguna ke folder tertentu

memberikan pengguna akses ke folder tertentu

Misalkan Anda mencoba memberikan pengguna akses ke folder tertentu. Jika pengguna IAM dan bucket S3 milik yang sama Akun AWS, maka Anda dapat menggunakan kebijakan IAM untuk memberikan akses pengguna ke folder bucket tertentu. Dengan pendekatan ini, Anda tidak perlu memperbarui kebijakan bucket untuk memberikan akses. Anda dapat menambahkan kebijakan IAM ke peran IAM yang dapat dialihkan oleh beberapa pengguna.

Jika identitas IAM dan bucket S3 milik yang berbeda Akun AWS, Anda harus memberikan akses lintas akun baik dalam kebijakan IAM maupun kebijakan bucket. Untuk informasi selengkapnya tentang pemberian akses lintas akun, lihat Pemilik bucket yang memberikan izin bucket lintas akun.

Contoh kebijakan bucket berikut memberikan akses konsol *JohnDoe* penuh hanya ke foldernya (home/*JohnDoe*/). Dengan membuat home folder dan memberikan izin yang sesuai kepada pengguna, Anda dapat meminta beberapa pengguna berbagi satu bucket. Kebijakan ini terdiri dari tiga `Allow` pernyataan:

- *AllowRootAndHomeListingOfCompanyBucket*: Memungkinkan user (*JohnDoe*) untuk daftar objek di tingkat root *DOC-EXAMPLE-BUCKET* bucket dan di home folder. Pernyataan ini juga memungkinkan pengguna untuk mencari pada prefiks home/ dengan menggunakan konsol.
- *AllowListingOfUserFolder*: Memungkinkan user (*JohnDoe*) untuk daftar semua objek dalam home/*JohnDoe*/ folder dan subfolder apapun.
- *AllowAllS3ActionsInUserFolder*: Memungkinkan pengguna untuk melakukan semua tindakan Amazon S3 dengan memberikan `Read`, `Write`, dan izin `Delete`. Izin terbatas pada folder beranda pemilik bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRootAndHomeListingOfCompanyBucket",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:user/JohnDoe"
                ]
            },
            "Effect": "Allow",
```

```
            "Action": ["s3:ListBucket"],
            "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
            "Condition": {
                "StringEquals": {
                    "s3:prefix": ["", "home/", "home/JohnDoe"],
                    "s3:delimiter": ["/"]
                }
            }
        },
        {
            "Sid": "AllowListingOfUserFolder",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:user/JohnDoe"
                ]
            },
            "Action": ["s3:ListBucket"],
            "Effect": "Allow",
            "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
            "Condition": {
                "StringLike": {
                    "s3:prefix": ["home/JohnDoe/*"]
                }
            }
        },
        {
            "Sid": "AllowAllS3ActionsInUserFolder",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:user/JohnDoe"
                ]
            },
            "Action": ["s3:*"],
            "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/home/JohnDoe/*"]
        }
    ]
}
```

Mengelola akses log akses

Berikan akses ke Penyeimbang Beban Aplikasi untuk mengaktifkan log akses

Saat Anda mengaktifkan pencatatan akses untuk Penyeimbang Beban Aplikasi, Anda harus menentukan nama bucket S3 tempat penyeimbang beban akan menyimpan log. Bucket harus memiliki kebijakan terlampir yang memberikan izin Elastic Load Balancing untuk menulis ke bucket.

Pada contoh berikut, kebijakan bucket memberikan izin Elastic Load Balancing (ELB) untuk menulis log akses ke bucket:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": {
        "AWS": "arn:aws:iam::elb-account-id:root"
      },
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::example-s3-bucket/prefix/AWSLogs/111122223333/*"
    }
  ]
}
```

> **Note**
>
> Pastikan untuk mengganti *elb-account-id* dengan Akun AWS ID untuk Elastic Load Balancing untuk Anda. Wilayah AWS Untuk daftar Wilayah Elastic Load Balancing, lihat Melampirkan kebijakan ke bucket Amazon S3 Anda di Panduan Pengguna Elastic Load Balancing.

Jika Anda Wilayah AWS tidak muncul di daftar Wilayah Elastic Load Balancing yang didukung, gunakan kebijakan berikut, yang memberikan izin ke layanan pengiriman log yang ditentukan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
       "Principal": {
```

```
          "Service": "logdelivery.elasticloadbalancing.amazonaws.com"
           },
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::example-s3-bucket/prefix/AWSLogs/111122223333/*"
    }
  ]
}
```

Kemudian, pastikan untuk mengonfigurasi log akses Elastic Load Balancing Anda dengan mengaktifkannya. Anda dapat memverifikasi izin bucket dengan membuat file pengujian.

Mengelola akses ke Amazon CloudFront OAI

Berikan izin ke Amazon CloudFront OAI

Contoh kebijakan bucket berikut memberikan izin identitas akses CloudFront asal (OAI) untuk mendapatkan (membaca) semua objek di bucket S3 Anda. Anda dapat menggunakan CloudFront OAI untuk memungkinkan pengguna mengakses objek di bucket Anda CloudFront tetapi tidak secara langsung melalui Amazon S3. Untuk informasi selengkapnya, lihat Membatasi akses ke konten Amazon S3 menggunakan Identitas Akses Asal di Panduan Pengembang CloudFront Amazon.

Kebijakan berikut menggunakan ID OAI sebagai `Principal`. Untuk informasi selengkapnya tentang menggunakan kebijakan bucket S3 guna memberikan akses ke CloudFront OAI, lihat Memigrasi dari identitas akses asal (OAI) ke kontrol akses asal (OAC) di Panduan Pengembang Amazon. CloudFront

Untuk menggunakan contoh ini:

- Ganti *`EH1HDMB1FH2TC`* dengan ID OAI. Untuk menemukan ID OAI, lihat halaman Origin Access Identity di CloudFront konsol, atau gunakan ListCloudFrontOriginAccessIdentitiesdi CloudFront API.
- Ganti *`example-s3-bucket`* dengan nama bucket Anda.

```
{
    "Version": "2012-10-17",
    "Id": "PolicyForCloudFrontPrivateContent",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Principal": {
                "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access
 Identity EH1HDMB1FH2TC"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::example-s3-bucket/*"
        }
    ]
}
```

Mengelola akses untuk Lensa Penyimpanan Amazon S3

Berikan Izin untuk Lensa Penyimpanan Amazon S3

Lensa Penyimpanan S3 menggabungkan metrik Anda dan menampilkan informasi di bagian Snapshot akun di halaman Bucket konsol Amazon S3. Lensa Penyimpanan S3 juga menyediakan dasbor interaktif yang dapat Anda gunakan untuk memvisualisasikan wawasan dan tren, menandai outlier, serta menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik perlindungan data. Dasbor Anda memiliki opsi perincian untuk menghasilkan dan memvisualisasikan wawasan di tingkat organisasi, akun, Wilayah AWS, kelas penyimpanan, bucket, prefiks, atau grup Lensa Penyimpanan. Anda juga dapat mengirimkan ekspor metrik harian dalam format CSV atau Parquet ke bucket S3.

Lensa Penyimpanan S3 dapat mengekspor metrik penggunaan penyimpanan gabungan Anda ke bucket Amazon S3 untuk analisis lebih lanjut. Bucket tempat Lensa Penyimpanan S3 menempatkan ekspor metrik nya dikenal sebagai bucket tujuan. Saat mengatur ekspor metrik Lensa Penyimpanan S3, Anda harus memiliki kebijakan bucket untuk bucket tujuan. Untuk informasi selengkapnya, lihat Menilai aktivitas penyimpanan dan penggunaan Anda dengan Amazon S3 Storage Lens.

Contoh kebijakan bucket berikut memberikan kepada Amazon S3 izin untuk tulis objek PUT (permintaan) ke bucket tujuan. Anda menggunakan kebijakan bucket seperti ini pada bucket tujuan saat membuat pengaturan ekspor metrik Lensa Penyimpanan S3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3StorageLensExamplePolicy",
            "Effect": "Allow",
            "Principal": {
                "Service": "storage-lens.s3.amazonaws.com"
```

```
            },
            "Action": "s3:PutObject",
            "Resource": [
                "arn:aws:s3:::destination-bucket/destination-prefix/
StorageLens/111122223333/*"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceAccount": "111122223333",
                    "aws:SourceArn": "arn:aws:s3:region-code:111122223333:storage-
lens/storage-lens-dashboard-configuration-id"
                }
            }
        }
    ]
}
```

Saat Anda menyiapkan ekspor metrik tingkat organisasi Lensa Penyimpanan S3, gunakan modifikasi berikut pada pernyataan `Resource` kebijakan bucket sebelumnya.

```
"Resource": "arn:aws:s3:::destination-bucket/destination-prefix/StorageLens/your-
organization-id/*",
```

Mengelola izin untuk laporan Inventaris S3, S3 analytics, dan Inventaris S3

Berikan izin untuk Inventaris S3 dan analitik S3

S3 Inventaris membuat daftar objek dalam bucket, dan ekspor Analisis Kelas Penyimpanan analitik S3 membuat file output dari data yang digunakan dalam analisis. Bucket tempat inventaris mencantumkan objek disebut bucket sumber. Bucket tempat file inventaris atau file ekspor analitik ditulis disebut bucket tujuan. Saat membuat inventaris atau ekspor analitik, Anda harus membuat kebijakan bucket untuk bucket tujuan. Untuk informasi selengkapnya, lihat Inventaris Amazon S3 dan Analitik Amazon S3–Analisis Kelas Penyimpanan.

Contoh kebijakan bucket berikut memberikan kepada Amazon S3 izin untuk menulis objek PUT (permintaan) dari akun untuk bucket sumber ke bucket tujuan. Anda menggunakan kebijakan bucket seperti ini pada bucket tujuan saat menyiapkan S3 Inventory, dan ekspor analitik S3.

```
{
```

```
        "Version": "2012-10-17",
        "Statement": [
          {
              "Sid": "InventoryAndAnalyticsExamplePolicy",
              "Effect": "Allow",
              "Principal": {
                  "Service": "s3.amazonaws.com"
              },
              "Action": "s3:PutObject",
              "Resource": [
                  "arn:aws:s3:::DOC-EXAMPLE-DESTINATION-BUCKET/*"
              ],
              "Condition": {
                  "ArnLike": {
                      "aws:SourceArn": "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET"
                  },
                  "StringEquals": {
                      "aws:SourceAccount": "111122223333",
                      "s3:x-amz-acl": "bucket-owner-full-control"
                  }
              }
          }
      ]
}
```

Kontrol pembuatan konfigurasi laporan Inventaris S3

Inventaris Amazon S3membuat daftar objek dalam bucket S3 dan metadata untuk setiap objek. `s3:PutInventoryConfiguration`Izin ini memungkinkan pengguna untuk membuat konfigurasi inventaris yang mencakup semua bidang metadata objek yang tersedia secara default dan menentukan bucket tujuan untuk menyimpan inventaris. Pengguna dengan akses baca ke objek di bucket tujuan dapat mengakses semua bidang metadata objek yang tersedia dalam laporan inventaris. Untuk informasi selengkapnya tentang bidang metadata yang tersedia di Inventaris S3, lihat. Daftar Inventaris Amazon S3

Untuk membatasi pengguna mengonfigurasi laporan Inventaris S3, hapus `s3:PutInventoryConfiguration` izin dari pengguna.

Beberapa bidang metadata objek dalam konfigurasi laporan Inventaris S3 bersifat opsional, artinya tersedia secara default tetapi dapat dibatasi saat Anda memberikan izin kepada pengguna. `s3:PutInventoryConfiguration` Anda dapat mengontrol apakah pengguna dapat menyertakan bidang metadata opsional ini dalam laporan mereka dengan menggunakan kunci

`s3:InventoryAccessibleOptionalFields` kondisi. Untuk daftar bidang metadata opsional yang tersedia di S3 Inventory, lihat OptionalFieldsdi Referensi API Amazon Simple Storage Service.

Untuk memberikan izin kepada pengguna untuk membuat konfigurasi inventaris dengan bidang metadata opsional tertentu, gunakan kunci `s3:InventoryAccessibleOptionalFields` kondisi untuk menyempurnakan kondisi dalam kebijakan bucket Anda.

Contoh kebijakan berikut memberikan izin kepada pengguna (*Ana*) untuk membuat konfigurasi inventaris secara kondisional. `ForAllValues:StringEquals`Kondisi dalam kebijakan menggunakan kunci `s3:InventoryAccessibleOptionalFields` kondisi untuk menentukan dua bidang metadata opsional yang diizinkan, yaitu `Size` dan. `StorageClass` Jadi, saat *Ana* membuat konfigurasi inventaris, satu-satunya bidang metadata opsional yang dapat dia sertakan adalah `Size` dan. `StorageClass`

```
{
 "Id": "InventoryConfigPolicy",
 "Version": "2012-10-17",
 "Statement": [{
   "Sid": "AllowInventoryCreationConditionally",
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:user/Ana"
   },
   "Action":
    "s3:PutInventoryConfiguration",
   "Resource":
    "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET",
   "Condition": {
    "ForAllValues:StringEquals": {
     "s3:InventoryAccessibleOptionalFields": [
        "Size",
        "StorageClass"
        ]
      }
    }
   }
 ]
}
```

Untuk membatasi pengguna mengonfigurasi laporan Inventaris S3 yang menyertakan bidang metadata opsional tertentu, tambahkan `Deny` pernyataan eksplisit ke kebijakan bucket untuk

bucket sumber. Contoh kebijakan bucket berikut menyangkal pengguna *Ana* membuat konfigurasi inventaris di bucket sumber *DOC-EXAMPLE-SOURCE-BUCKET* yang menyertakan bidang opsional `ObjectAccessControlList` atau `ObjectOwner` metadata. Pengguna masih *Ana* dapat membuat konfigurasi inventaris dengan bidang metadata opsional lainnya.

```
{
 "Id": "InventoryConfigSomeFields",
 "Version": "2012-10-17",
 "Statement": [{
   "Sid": "AllowInventoryCreation",
   "Effect": "Allow",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:user/Ana"
   },
   "Action": "s3:PutInventoryConfiguration",
   "Resource":
    "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET",

  },
  {
   "Sid": "DenyCertainInventoryFieldCreation",
   "Effect": "Deny",
   "Principal": {
    "AWS": "arn:aws:iam::111122223333:user/Ana"
   },
   "Action": "s3:PutInventoryConfiguration",
   "Resource":
     "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET",
   "Condition": {
    "ForAnyValue:StringEquals": {
     "s3:InventoryAccessibleOptionalFields": [
        "ObjectOwner",
        "ObjectAccessControlList"
        ]
      }
    }
   }
 ]
}
```

> **Note**
>
> Penggunaan kunci `s3:InventoryAccessibleOptionalFields` kondisi dalam kebijakan bucket tidak memengaruhi pengiriman laporan inventaris berdasarkan konfigurasi inventaris yang ada.

> **Important**
>
> Kami menyarankan Anda menggunakan `ForAllValues` dengan `Allow` efek atau `ForAnyValue` `Deny` efek, seperti yang ditunjukkan pada contoh sebelumnya. Jangan gunakan `ForAllValues` dengan `Deny` efek atau `ForAnyValue` `Allow` efek, karena kombinasi ini bisa terlalu membatasi dan memblokir penghapusan konfigurasi inventaris. Untuk mempelajari selengkapnya tentang operator set `ForAllValues` dan `ForAnyValue` kondisi, lihat Kunci konteks Multivalued di Panduan Pengguna IAM.

Membutuhkan MFA

Amazon S3 mendukung akses API yang dilindungi MFA, sebuah fitur yang dapat memberlakukan autentikasi multi-faktor (MFA) untuk mengakses sumber daya Amazon S3. Otentikasi multi-faktor memberikan tingkat keamanan ekstra yang dapat Anda terapkan ke lingkungan Anda AWS . MFA adalah fitur keamanan yang mengharuskan pengguna membuktikan kepemilikan fisik perangkat MFA dengan menyediakan kode MFA yang valid. Untuk informasi selengkapnya, lihat Autentikasi Multi-Faktor AWS. Anda dapat mewajibkan MFA untuk setiap permintaan akses sumber daya Amazon S3.

Untuk memberlakukan persyaratan MFA, gunakan kunci kondisi `aws:MultiFactorAuthAge` dalam kebijakan bucket. Pengguna IAM dapat mengakses sumber daya Amazon S3 dengan menggunakan kredensyal sementara yang dikeluarkan oleh (). AWS Security Token Service AWS STS Anda memberikan kode MFA pada saat permintaan AWS STS .

Saat Amazon S3 menerima permintaan dengan autentikasi multi-faktor, kunci `aws:MultiFactorAuthAge` kondisi memberikan nilai numerik yang menunjukkan berapa lama (dalam detik) kredensial sementara dibuat. Jika kredensial sementara yang diberikan dalam permintaan tidak dibuat menggunakan perangkat MFA, maka nilai kunci ini akan kosong (tidak ada). Dalam kebijakan bucket, Anda dapat menambahkan sebuah kondisi untuk memeriksa nilai ini, seperti yang ditunjukkan dalam contoh berikut.

Contoh kebijakan ini menolak operasi Amazon S3 pada `/taxdocuments`folder dalam bucket jika permintaan tidak `example-s3-bucket` diautentikasi dengan menggunakan MFA. Untuk pelajari selengkapnya tentang otentikasi multifaktor (MFA), lihat Menggunakan Autentikasi Multi-Faktor (MFA) di AWS dalam Panduan Pengguna IAM.

```
{
    "Version": "2012-10-17",
    "Id": "123",
    "Statement": [
      {
        "Sid": "",
        "Effect": "Deny",
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::example-s3-bucket/taxdocuments/*",
        "Condition": { "Null": { "aws:MultiFactorAuthAge": true }}
      }
    ]
 }
```

`Null`Kondisi di `Condition` blok mengevaluasi `true` apakah nilai kunci `aws:MultiFactorAuthAge` kondisi adalah nol, yang mana hal ini menunjukkan bahwa kredensial keamanan sementara dalam permintaan dibuat tanpa perangkat MFA.

Kebijakan bucket berikut merupakan perpanjangan dari kebijakan bucket sebelumnya. Kebijakan bucket berikut mencakup dua pernyataan kebijakan. Satu pernyataan memberikan izin `s3:GetObject` pada bucket (`example-s3-bucket`) kepada semua orang. Pernyataan lain membatasi lebih lanjut akses ke folder `example-s3-bucket/taxdocuments` yang ada dalam bucket dengan mewajibkan MFA.

```
{
    "Version": "2012-10-17",
    "Id": "123",
    "Statement": [
      {
        "Sid": "",
        "Effect": "Deny",
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::example-s3-bucket/taxdocuments/*",
        "Condition": { "Null": { "aws:MultiFactorAuthAge": true } }
```

```
      },
      {
        "Sid": "",
        "Effect": "Allow",
        "Principal": "*",
        "Action": ["s3:GetObject"],
        "Resource": "arn:aws:s3:::example-s3-bucket/*"
      }
    ]
 }
```

Anda dapat menggunakan kondisi numerik secara opsional untuk membatasi durasi berlakunya `aws:MultiFactorAuthAge` kunci. Durasi yang Anda tentukan dengan `aws:MultiFactorAuthAge` kunci tidak bergantung pada usia pakai kredensial keamanan sementara yang digunakan untuk mengautentikasi permintaan.

Sebagai contoh, kebijakan bucket berikut, selain mewajibkan autentikasi MFA, juga memeriksa berapa lama lagi sesi sementara dibuat. Kebijakan tersebut menolak setiap operasi jika nilai kunci `aws:MultiFactorAuthAge` menunjukkan bahwa sesi sementara dibuat lebih dari satu jam yang lalu (3.600 detik).

```
{
    "Version": "2012-10-17",
    "Id": "123",
    "Statement": [
      {
        "Sid": "",
        "Effect": "Deny",
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::example-s3-bucket/taxdocuments/*",
        "Condition": {"Null": {"aws:MultiFactorAuthAge": true }}
      },
      {
        "Sid": "",
        "Effect": "Deny",
        "Principal": "*",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::example-s3-bucket/taxdocuments/*",
        "Condition": {"NumericGreaterThan": {"aws:MultiFactorAuthAge": 3600 }}
       },
       {
```

```
        "Sid": "",
        "Effect": "Allow",
        "Principal": "*",
        "Action": ["s3:GetObject"],
        "Resource": "arn:aws:s3:::example-s3-bucket/*"
      }
    ]
}
```

Mencegah pengguna menghapus objek

Secara bawaan, pengguna tidak memiliki izin. Namun saat Anda membuat kebijakan, Anda mungkin memberikan izin kepada pengguna yang tidak ingin Anda berikan. Untuk menghindari celah izin tersebut, Anda dapat menulis kebijakan akses yang lebih ketat dengan menambahkan penolakan eksplisit.

Untuk secara eksplisit memblokir pengguna atau akun agar tidak menghapus objek, Anda harus menambahkan tindakan berikut ke kebijakan bucket:`s3:DeleteObject`,`s3:DeleteObjectVersion`, dan izin. `s3:PutLifecycleConfiguration` Ketiga tindakan tersebut diperlukan karena Anda dapat menghapus objek baik dengan memanggil DELETE Object API secara eksplisit atau dengan mengonfigurasi siklus hidupnya (lihat) Mengelola siklus hidup penyimpanan Anda sehingga Amazon S3 dapat menghapus objek saat masa pakainya berakhir.

Dalam contoh kebijakan berikut, Anda secara eksplisit menolak izin DELETE Object ke pengguna Dave. Penolakan eksplisit selalu menggantikan izin lain yang diberikan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "statement1",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/Dave"
      },
      "Action": [
        "s3:GetObjectVersion",
        "s3:GetBucketAcl"
      ],
      "Resource": [
```

```
        "arn:aws:s3:::example-s3-bucket1",
    "arn:aws:s3:::example-s3-bucket1/*"
      ]
    },
    {
      "Sid": "statement2",
      "Effect": "Deny",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/Dave"
      },
      "Action": [
        "s3:DeleteObject",
        "s3:DeleteObjectVersion",
        "s3:PutLifecycleConfiguration"
      ],
      "Resource": [
        "arn:aws:s3:::example-s3-bucket1",
     "arn:aws:s3:::example-s3-bucket1/*"
      ]
    }
  ]
}
```

Contoh kebijakan bucket menggunakan tombol kondisi

Anda dapat menggunakan bahasa kebijakan akses untuk menentukan syarat-syarat saat memberikan izin. Anda dapat menggunakan elemen `Condition`, atau blok `Condition` tambahan untuk menentukan syarat saat kebijakan berlaku.

Untuk kebijakan yang menggunakan kunci syarat Amazon S3 untuk operasi objek dan bucket, lihat contoh berikut. Untuk informasi lebih lanjut tentang kunci syarat ini, lihat Kunci kondisi kebijakan untuk Amazon S3. Untuk daftar lengkap tindakan Amazon S3, kunci kondisi, dan sumber daya yang dapat Anda tentukan dalam kebijakan, lihat Tindakan, sumber daya, dan kunci kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Contoh—Kunci syarat Amazon S3 untuk operasi objek

Bagian ini memberikan contoh-contoh yang menunjukkan cara menggunakan kunci kondisi spesifik Amazon S3 untuk operasi objek. Untuk daftar lengkap tindakan Amazon S3, kunci kondisi, dan sumber daya yang dapat Anda tentukan dalam kebijakan, lihat Tindakan, sumber daya, dan kunci kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Beberapa contoh kebijakan menunjukkan bagaimana Anda dapat menggunakan kunci kondisi operasi PUT Objek. Operasi PUT Objek memungkinkan header spesifik daftar kontrol akses (ACL) yang dapat Anda gunakan untuk memberikan izin berbasis ACL. Dengan menggunakan kunci ini, pemilik bucket dapat menetapkan kondisi-kondisi untuk mengharuskan izin akses kustom ketika pengguna mengunggah objek. Anda juga dapat memberikan izin berbasis ACL dengan operasi tersebut. PutObjectAcl Untuk informasi selengkapnya, lihat PutObjectAcldi Referensi API Layanan Penyimpanan Sederhana Amazon S3 Amazon. Untuk informasi selengkapnya tentang ACL, lihat Gambaran umum daftar kontrol akses (ACL).

Topik

- Contoh 1: Pemberian s3: PutObject izin yang membutuhkan objek yang disimpan menggunakan enkripsi sisi server
- Contoh 2: Pemberian s3: PutObject izin untuk menyalin objek dengan batasan pada sumber salinan
- Contoh 3: Memberikan akses ke versi objek tertentu
- Contoh 4: Memberikan izin berdasarkan tag objek
- Contoh 5: Membatasi akses oleh Akun AWS ID pemilik bucket
- Contoh 6: Membutuhkan versi TLS minimum

Contoh 1: Pemberian s3: PutObject izin yang membutuhkan objek yang disimpan menggunakan enkripsi sisi server

Misalkan ketika Akun A memiliki bucket. Administrator akun ingin memberikan Jane, pengguna di Akun A, izin untuk mengunggah objek dengan syarat Jane selalu meminta enkripsi server side sehingga Amazon S3 dapat menyimpan objek yang dienkripsi. Administrator Akun A dapat melakukannya dengan menggunakan kunci kondisi `s3:x-amz-server-side-encryption` seperti yang ditunjukkan. Pasangan nilai kunci dalam pemblokiran `Condition` menentukan kunci `s3:x-amz-server-side-encryption`.

```
"Condition": {
     "StringNotEquals": {
         "s3:x-amz-server-side-encryption": "AES256"
     }}
```

Saat menguji izin menggunakan AWS CLI, Anda harus menambahkan parameter yang diperlukan menggunakan `--server-side-encryption` parameter.

```
aws s3api put-object --bucket example1bucket --key HappyFace.jpg --body c:
\HappyFace.jpg --server-side-encryption "AES256" --profile AccountBadmin
```

Contoh 2: Pemberian s3: PutObject izin untuk menyalin objek dengan batasan pada sumber salinan

Dalam permintaan PUT Object, ketika Anda menentukan objek sumber, ini adalah operasi penyalinan (lihat PUT Objek-Salin). Oleh karena itu, pemilik bucket dapat memberikan izin kepada pengguna untuk menyalin objek dengan pembatasan pada sumber, misalnya:

- Izinkan penyalinan objek hanya dari bucket `sourcebucket`.
- Izinkan penyalinan objek dari bucket sumber dan hanya objek yang prefiks nama kuncinya dimulai dengan publik/ f (misalnya, sourcebucket/public/*).
- Izinkan penyalinan hanya pada objek tertentu dari sourcebucket (misalnya, sourcebucket/example.jpg).

Kebijakan bucket berikut ini memberikan pengguna (Dave) izin `s3:PutObject`. Itu memungkinkan dia untuk menyalin objek hanya dengan syarat bahwa permintaan tersebut menyertakan header `s3:x-amz-copy-source` dan nilai header menyebutkan prefiks nama kunci `/awsexamplebucket1/public/*`.

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
            "Sid": "cross-account permission to user in your own account",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/Dave"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::awsexamplebucket1/*"
        },
        {
            "Sid": "Deny your user permission to upload object if copy source is not /
bucket/folder",
            "Effect": "Deny",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/Dave"
            },
            "Action": "s3:PutObject",
```

```
            "Resource": "arn:aws:s3:::awsexamplebucket1/*",
            "Condition": {
                "StringNotLike": {
                    "s3:x-amz-copy-source": "awsexamplebucket1/public/*"
                }
            }
        }
    ]
}
```

Uji kebijakan dengan AWS CLI

Anda dapat menguji izin menggunakan AWS CLI `copy-object` perintah. Anda menentukan sumber dengan menambahkan parameter `--copy-source`; prefiks nama kunci harus cocok dengan prefiks yang diperbolehkan dalam kebijakan. Anda perlu memberikan kredensial Dave kepada pengguna dengan menggunakan parameter `--profile`. Untuk informasi selengkapnya tentang pengaturan AWS CLI, lihatMengembangkan dengan Amazon S3 menggunakan AWS CLI.

```
aws s3api copy-object --bucket awsexamplebucket1 --key HappyFace.jpg
--copy-source examplebucket/public/PublicHappyFace1.jpg --profile AccountADave
```

Berikan izin untuk menyalin objek tertentu

Kebijakan sebelumnya menggunakan kondisi `StringNotLike`. Untuk memberikan izin menyalin hanya objek tertentu, Anda harus mengubah kondisi dari `StringNotLike` menjadi `StringNotEquals` lalu tentukan kunci objek yang tepat seperti yang ditunjukkan.

```
"Condition": {
       "StringNotEquals": {
           "s3:x-amz-copy-source": "awsexamplebucket1/public/PublicHappyFace1.jpg"
       }
}
```

Contoh 3: Memberikan akses ke versi objek tertentu

Misalkan ketika Akun A memiliki bucket dengan versi aktif. Bucket tersebut memiliki beberapa versi objek `HappyFace.jpg`. Administrator akun sekarang ingin memberikan izin kepada penggunanya, yakni Dave, untuk mendapatkan versi objek tertentu saja. Administrator akun dapat melakukan hal ini dengan memberikan izin `s3:GetObjectVersion` kepada Dave dengan syarat seperti yang ditunjukkan di bawah ini. Pasangan nilai kunci dalam pemblokiran `Condition` menentukan kunci

kondisi `s3:VersionId`. Dalam kasus ini, Dave perlu mengetahui ID dari versi objek yang tepat untuk mengambil objek tersebut.

Untuk informasi selengkapnya, lihat GetObjectdi Referensi API Amazon Simple Storage Service.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "statement1",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/Dave"
            },
            "Action": "s3:GetObjectVersion",
            "Resource": "arn:aws:s3:::examplebucketversionenabled/HappyFace.jpg"
        },
        {
            "Sid": "statement2",
            "Effect": "Deny",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/Dave"
            },
            "Action": "s3:GetObjectVersion",
            "Resource": "arn:aws:s3:::examplebucketversionenabled/HappyFace.jpg",
            "Condition": {
                "StringNotEquals": {
                    "s3:VersionId": "AaaHbAQitwiL_h47_44lRO2DDfLlBO5e"
                }
            }
        }
    ]
}
```

Uji kebijakan dengan AWS CLI

Anda dapat menguji izin menggunakan AWS CLI `get-object` perintah dengan `--version-id` parameter yang mengidentifikasi versi objek tertentu. Perintah mengambil objek dan menyimpannya ke file `OutputFile.jpg`.

```
aws s3api get-object --bucket examplebucketversionenabled --key HappyFace.jpg
 OutputFile.jpg --version-id AaaHbAQitwiL_h47_44lRO2DDfLlBO5e --profile AccountADave
```

Contoh 4: Memberikan izin berdasarkan tag objek

Untuk contoh tentang cara menggunakan kunci kondisi penandaan objek dengan operasi Amazon S3, lihat Kebijakan pemberian tag dan kontrol akses.

Contoh 5: Membatasi akses oleh Akun AWS ID pemilik bucket

Anda dapat menggunakan kunci `aws:ResourceAccount` atau `s3:ResourceAccount` untuk menulis kebijakan titik akhir IAM atau cloud privat virtual (VPC) yang membatasi akses pengguna, peran, atau akses aplikasi ke bucket Amazon S3 yang dimiliki oleh ID Akun AWS tertentu. Gunakan kunci syarat ini untuk mencegah klien dalam VPC Anda mengakses bucket yang tidak Anda miliki.

Namun, ketahuilah bahwa beberapa AWS layanan bergantung pada akses ke bucket yang AWS dikelola. Oleh karena itu, menggunakan kunci `aws:ResourceAccount` atau `s3:ResourceAccount` dalam kebijakan IAM Anda juga dapat memengaruhi akses ke sumber daya ini.

Untuk informasi selengkapnya dan contoh, lihat sumber daya berikut ini:

- Batasi akses ke bucket dalam Akun AWS yang ditentukan dalam Panduan AWS PrivateLink
- Batasi akses ke bucket yang digunakan Amazon ECR dalam Panduan Amazon ECR
- Menyediakan akses yang diperlukan ke Systems Manager untuk bucket Amazon S3 AWS terkelola dalam Panduan AWS Systems Manager
- Batasi akses ke bucket Amazon S3 yang dimiliki oleh spesifik Akun AWS di Blog Penyimpanan AWS

Contoh 6: Membutuhkan versi TLS minimum

Anda dapat menggunakan kunci TlsVersion kondisi s3: untuk menulis IAM, Virtual Private Cloud Endpoint (VPCE), atau kebijakan bucket yang membatasi akses pengguna atau aplikasi ke bucket Amazon S3 berdasarkan versi TLS yang digunakan oleh klien. Anda dapat menggunakan kunci syarat ini untuk tulis kebijakan yang memerlukan versi TLS minimum.

Example

Contoh kebijakan bucket ini menolak PutObject permintaan klien yang memiliki versi TLS lebih rendah dari 1.2, misalnya 1.1 atau 1.0.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": [
                "arn:aws:s3:::example-s3-bucket1",
                "arn:aws:s3:::example-s3-bucket1/*"
            ],
            "Condition": {
                "NumericLessThan": {
                    "s3:TlsVersion": 1.2
                }
            }
        }
    ]
}
```

Example

Contoh kebijakan bucket ini memungkinkan PutObject permintaan oleh klien yang memiliki versi TLS lebih tinggi dari 1.1, misalnya 1.2, 1.3 atau lebih tinggi.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": [
                "arn:aws:s3:::example-s3-bucket1",
                "arn:aws:s3:::example-s3-bucket1/*"
            ],
            "Condition": {
                "NumericGreaterThan": {
                    "s3:TlsVersion": 1.1
                }
            }
        }
```

```
    ]
}
```

Contoh—Kunci syarat Amazon S3 untuk operasi bucket

Bagian ini memberikan contoh kebijakan yang menunjukkan bagaimana Anda dapat menggunakan kunci kondisi spesifik Amazon S3 untuk operasi bucket.

Topik

- Contoh 1: Pemberian s3: GetObject izin dengan kondisi pada alamat IP
- Contoh 2: Mendapatkan daftar objek di bucket dengan prefiks tertentu
- Contoh 3: Mengatur jumlah kunci maksimum

Contoh 1: Pemberian s3: GetObject izin dengan kondisi pada alamat IP

Anda dapat memberikan izin kepada pengguna yang diautentikasi untuk menggunakan `s3:GetObject` tindakan jika permintaan berasal dari rentang alamat IP tertentu (192.0.2.*), kecuali alamat IP 192.0.2.188. Dalam pemblokiran kondisi, `IpAddress` dan `NotIpAddress` adalah kondisi, dan setiap kondisi diberikan pasangan nilai kunci untuk evaluasi. Kedua pasangan kunci-nilai dalam contoh ini menggunakan kunci `aws:SourceIp` AWS-wide.

> **Note**
>
> Nilai kunci `IPAddress` dan `NotIpAddress` yang ditentukan dalam kondisi tersebut menggunakan notasi CIDR seperti yang dijelaskan dalam RFC 4632. Untuk informasi lebih lanjut, lihat http://www.rfc-editor.org/rfc/rfc4632.txt.

```
{
    "Version": "2012-10-17",
    "Id": "S3PolicyId1",
    "Statement": [
        {
            "Sid": "statement1",
            "Effect": "Allow",
            "Principal": "*",
            "Action":"s3:GetObject",
            "Resource": "arn:aws:s3:::awsexamplebucket1/*",
```

```
            "Condition" : {
                "IpAddress" : {
                    "aws:SourceIp": "192.0.2.0/24"
                },
                "NotIpAddress" : {
                    "aws:SourceIp": "192.0.2.188/32"
                }
            }
        }
    ]
}
```

Anda juga dapat menggunakan kunci kondisi AWS-wide lainnya dalam kebijakan Amazon S3. Misalnya, Anda dapat menentukan kunci kondisi `aws:SourceVpce` dan `aws:SourceVpc` dalam kebijakan bucket untuk titik akhir VPC. Untuk contoh spesifik, lihatMegendalikan akses dari titik akhir VPC dengan kebijakan bucket.

> **Note**
>
> Untuk beberapa kunci kondisi AWS global, hanya jenis sumber daya tertentu yang didukung. Oleh karena itu, periksa apakah Amazon S3 mendukung kunci kondisi global dan jenis sumber daya yang ingin Anda gunakan, atau apakah Anda perlu menggunakan kunci kondisi kustom Amazon S3. Untuk daftar lengkap jenis sumber daya dan kunci kondisi yang didukung untuk Amazon S3, lihat Tindakan, sumber daya, dan kunci kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Contoh 2: Mendapatkan daftar objek di bucket dengan prefiks tertentu

Anda dapat menggunakan kunci `s3:prefix` kondisi untuk membatasi respons GET Bucket (ListObjects) API ke nama kunci dengan awalan tertentu. Apabila Anda adalah pemilik bucket, Anda dapat membatasi pengguna untuk mencantumkan konten prefiks tertentu yang ada dalam bucket. Kunci kondisi ini berguna jika objek dalam bucket diatur berdasarkan prefiks nama kunci. Konsol Amazon S3 menggunakan prefiks nama kunci untuk menampilkan konsep folder. Hanya konsol tersebut yang mendukung konsep folder; Amazon S3 API hanya mendukung bucket dan objek. Untuk informasi lebih lanjut tentang bagaimana menggunakan prefiks dan pembatas untuk memfilter izin akses, lihat Mengontrol akses ke bucket dengan kebijakan pengguna.

Sebagai contoh, jika Anda memiliki dua objek dengan nama kunci `public/object1.jpg` dan `public/object2.jpg`, maka konsol akan menunjukkan objek di bawah folder `public`. Pada

Amazon S3 API, objek-objek ini adalah objek dengan prefiks, bukan objek yang ada dalam folder. Namun, pada Amazon S3 API, jika Anda mengorganisir kunci objek Anda dengan menggunakan prefiks tersebut, maka Anda dapat memberikan izin `s3:ListBucket` dengan kondisi `s3:prefix` yang memungkinkan pengguna mendapatkan daftar nama kunci yang memiliki prefiks tertentu tersebut.

Dalam contoh ini, pemilik bucket dan akun induk yang merupakan pemilik dari pengguna adalah sama. Jadi, pemilik bucket dapat menggunakan kebijakan bucket atau kebijakan pengguna. Untuk informasi selengkapnya tentang kunci kondisi lain yang dapat Anda gunakan dengan GET Bucket (ListObjects) API, lihat ListObjects.

Kebijakan pengguna

Kebijakan pengguna berikut memberikan izin `s3:ListBucket` (lihat GET Bucket (List Objects)) dengan kondisi yang mengharuskan pengguna untuk menentukan `prefix` dalam permintaan dengan nilai `projects`.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Action": "s3:ListBucket",
         "Resource":"arn:aws:s3:::awsexamplebucket1",
         "Condition" : {
             "StringEquals" : {
                 "s3:prefix": "projects"
             }
          }
       },
      {
         "Sid":"statement2",
         "Effect":"Deny",
         "Action": "s3:ListBucket",
         "Resource": "arn:aws:s3:::awsexamplebucket1",
         "Condition" : {
             "StringNotEquals" : {
                 "s3:prefix": "projects"
             }
          }
       }
```

```
    ]
}
```

Kondisi ini membatasi pengguna untuk mendaftarkan kunci objek dengan prefiks `projects`. Penolakan secara jelas tambahan menolak permintaan pengguna untuk mencantumkan kunci dengan prefiks lain, terlepas dari izin lain apa pun yang mungkin dimiliki oleh pengguna. Sebagai contoh, pengguna mungkin mendapatkan izin untuk mencantumkan kunci objek tanpa pembatasan apa pun, baik dengan pembaruan terhadap kebijakan pengguna sebelumnya atau melalui kebijakan bucket. Karena penolakan secara jelas selalu dapat menggantikan, permintaan pengguna untuk mencantumkan kunci selain prefiks `projects` ditolak.

Kebijakan bucket

Jika Anda menambahkan elemen `Principal` dalam kebijakan pengguna di atas, yang mengidentifikasi pengguna, sekarang Anda memiliki kebijakan bucket seperti yang ditunjukkan.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/bucket-owner"
         },
         "Action":  "s3:ListBucket",
         "Resource": "arn:aws:s3:::awsexamplebucket1",
         "Condition" : {
             "StringEquals" : {
                 "s3:prefix": "projects"
             }
          }
       },
      {
         "Sid":"statement2",
         "Effect":"Deny",
         "Principal": {
            "AWS": "arn:aws:iam::123456789012:user/bucket-owner"
         },
         "Action": "s3:ListBucket",
         "Resource": "arn:aws:s3:::awsexamplebucket1",
         "Condition" : {
```

```
            "StringNotEquals" : {
                "s3:prefix": "projects"
            }
        }
    }
  ]
}
```

Uji kebijakan dengan AWS CLI

Anda dapat menguji kebijakan menggunakan `list-object` AWS CLI perintah berikut. Dalam perintah ini, Anda memberikan kredensial pengguna dengan menggunakan parameter `--profile`. Untuk informasi selengkapnya tentang pengaturan dan penggunaan AWS CLI, lihatMengembangkan dengan Amazon S3 menggunakan AWS CLI.

```
aws s3api list-objects --bucket awsexamplebucket1 --prefix examplefolder --profile
 AccountADave
```

Apabila bucket tersebut merupakan versi-diaktifkan, untuk mencantumkan objek dalam bucket, Anda harus memberikan izin `s3:ListBucketVersions` dalam kebijakan sebelumnya, alih-alih izin `s3:ListBucket`. Izin ini turut mendukung kunci kondisi `s3:prefix`.

Contoh 3: Mengatur jumlah kunci maksimum

Anda dapat menggunakan tombol `s3:max-keys` kondisi untuk mengatur jumlah maksimum kunci yang dapat dikembalikan oleh pemohon dalam GET Bucket (ListObjects) atau ListObjectVersionspermintaan. Secara bawaan, API akan mengembalikan hingga 1.000 kunci. Untuk daftar operator kondisi numerik yang dapat Anda gunakan dengan `s3:max-keys` dan contoh yang menyertainya, lihat Operator kondisi Numerik dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas untuk Amazon S3

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau memodifikasi sumber daya Amazon S3. Mereka juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS API. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian akan dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM menggunakan contoh dokumen kebijakan JSON ini, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Untuk detail tentang tindakan dan jenis sumber daya yang ditentukan oleh Amazon S3, termasuk format ARN untuk setiap jenis sumber daya, lihat Tindakan, sumber daya, dan kunci kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Topik

- Praktik terbaik kebijakan
- Contoh kebijakan berbasis identitas untuk Amazon S3
- Mengontrol akses ke bucket dengan kebijakan pengguna

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus sumber daya Amazon S3 di akun Anda. Tindakan ini membuat Akun AWS Anda dikenai biaya. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit — Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Anda Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat Kebijakan yang dikelola AWS atau Kebijakan yang dikelola AWS untuk fungsi tugas dalam Panduan Pengguna IAM.
- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk mengajukan izin, lihat Kebijakan dan izin dalam IAM dalam Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui yang spesifik Layanan AWS, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat Elemen kebijakan JSON IAM: Kondisi dalam Panduan Pengguna IAM.

- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat Validasi kebijakan IAM Access Analyzer dalam Panduan Pengguna IAM.
- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan. Untuk meminta MFA ketika operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi akses API yang dilindungi MFA dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat Praktik terbaik keamanan dalam IAM dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk Amazon S3

Bagian ini menunjukkan beberapa contoh kebijakan berbasis identitas AWS Identity and Access Management (IAM) untuk mengontrol akses ke Amazon S3. Misalnya kebijakan bucket (kebijakan berbasis sumber daya), lihat. Kebijakan bucket untuk Amazon S3 Untuk informasi lebih lanjut tentang bahasa kebijakan IAM, lihatKebijakan dan izin di Amazon S3.

Contoh kebijakan berikut akan berfungsi jika Anda menggunakannya secara terprogram. Tetapi, untuk menggunakannya dengan konsol Amazon S3, Anda harus memberikan izin tambahan yang diperlukan oleh konsol. Untuk informasi tentang menggunakan kebijakan seperti ini dengan konsol Amazon S3, lihat Mengontrol akses ke bucket dengan kebijakan pengguna.

Topik

- Mengizinkan akses pengguna IAM ke salah satu bucket Anda
- Mengizinkan setiap akses pengguna IAM ke folder dalam bucket
- Mengizinkan grup memiliki folder bersama di Amazon S3
- Mengizinkan semua pengguna Anda membaca objek di sebagian bucket
- Mengizinkan mitra untuk menempatkan file ke dalam bagian tertentu bucket
- Membatasi akses ke bucket Amazon S3 dalam bucket tertentu Akun AWS
- Membatasi akses ke bucket Amazon S3 dalam unit organisasi Anda

- Membatasi akses ke bucket Amazon S3 di organisasi Anda
- Memberikan izin untuk mengambil PublicAccessBlock konfigurasi untuk Akun AWS
- Membatasi pembuatan bucket ke satu Wilayah

Mengizinkan akses pengguna IAM ke salah satu bucket Anda

Dalam contoh ini, Anda ingin memberikan pengguna IAM dalam Akun AWS akses ke salah satu bucket Anda, *example-s3-bucket1*, dan mengizinkan pengguna untuk menambahkan, memperbarui, dan menghapus objek.

Selain memberikan izin `s3:PutObject`, `s3:GetObject`, dan `s3:DeleteObject` bagi pengguna, kebijakan tersebut juga memberikan izin `s3:ListAllMyBuckets`, `s3:GetBucketLocation`, dan `s3:ListBucket`. Izin-izin tersebut adalah izin tambahan yang diperlukan oleh konsol tersebut. Selain itu, tindakan `s3:PutObjectAcl` dan `s3:GetObjectAcl` diperlukan untuk dapat menyalin, memotong, dan menempel objek di konsol. Untuk panduan contoh yang memberikan izin kepada pengguna dan mengujinya dengan menggunakan konsol, lihat Mengontrol akses ke bucket dengan kebijakan pengguna.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action": "s3:ListAllMyBuckets",
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":["s3:ListBucket","s3:GetBucketLocation"],
         "Resource":"arn:aws:s3:::example-s3-bucket1"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:GetObject",
            "s3:GetObjectAcl",
            "s3:DeleteObject"
         ],
```

```
            "Resource":"arn:aws:s3:::example-s3-bucket1/*"
         }
      ]
   }
```

Mengizinkan setiap akses pengguna IAM ke folder dalam bucket

Dalam contoh ini, Anda ingin dua pengguna IAM, Mary dan Carlos, memiliki akses ke bucket Anda, *example-s3-bucket1, sehingga mereka dapat menambahkan, memperbarui*, dan menghapus objek. Namun, Anda ingin membatasi setiap akses pengguna ke satu prefiks (folder) dalam bucket. Anda dapat membuat folder dengan nama yang cocok dengan nama pengguna mereka.

```
example-s3-bucket1
   Mary/
   Carlos/
```

Untuk memberikan setiap pengguna akses hanya ke foldernya saja, Anda dapat menulis kebijakan untuk setiap pengguna dan melampirkannya secara sendiri-sendiri. Misalnya, Anda dapat melampirkan kebijakan berikut ke pengguna Mary untuk mengizinkan izin Amazon S3 tertentu pada folder *example-s3-bucket1*/*Mary*.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::example-s3-bucket1/Mary/*"
      }
   ]
}
```

Anda kemudian dapat melampirkan kebijakan serupa ke pengguna Carlos, menentukan folder *Carlos* dalam nilai`Resource`.

Alih-alih melampirkan kebijakan kepada masing-masing pengguna, Anda dapat menulis satu kebijakan yang menggunakan variabel kebijakan dan melampirkan kebijakan tersebut ke grup. Pertama, Anda harus membuat grup dan menambahkan Mary dan Carlos ke grup. Kebijakan contoh berikut memungkinkan serangkaian izin Amazon S3 pada folder *example-s3-bucket1*/${aws:username}. Ketika kebijakan dievaluasi, variabel ${aws:username} kebijakan diganti dengan nama pengguna pemohon. Misalnya, jika Mary mengirim permintaan untuk menempatkan objek, maka operasi hanya diperbolehkan jika Mary mengunggah objek tersebut ke *example-s3-bucket1*/Mary folder.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::example-s3-bucket1/${aws:username}/*"
      }
   ]
}
```

> **Note**
>
> Saat menggunakan variabel kebijakan, Anda harus secara jelas menyebutkan versi 2012-10-17 dalam kebijakan tersebut. Versi bawaan bahasa kebijakan IAM, yaitu 2008-10-17, tidak mendukung variabel kebijakan ini.

Jika Anda ingin menguji kebijakan sebelumnya di konsol Amazon S3, maka konsol tersebut memerlukan izin tambahan, seperti yang ditunjukkan dalam kebijakan berikut. Untuk informasi tentang bagaimana konsol menggunakan izin ini, lihat Mengontrol akses ke bucket dengan kebijakan pengguna.

```
{
 "Version":"2012-10-17",
```

```
  "Statement": [
    {
      "Sid": "AllowGroupToSeeBucketListInTheConsole",
      "Action": [
       "s3:ListAllMyBuckets",
       "s3:GetBucketLocation"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Sid": "AllowRootLevelListingOfTheBucket",
      "Action": "s3:ListBucket",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::example-s3-bucket1",
      "Condition":{
            "StringEquals":{
                    "s3:prefix":[""], "s3:delimiter":["/"]
                           }
                }
    },
    {
      "Sid": "AllowListBucketOfASpecificUserPrefix",
      "Action": "s3:ListBucket",
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::example-s3-bucket1",
      "Condition":{  "StringLike":{"s3:prefix":["${aws:username}/*"] }
       }
    },
      {
     "Sid": "AllowUserSpecificActionsOnlyInTheSpecificUserPrefix",
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::example-s3-bucket1/${aws:username}/*"
      }
  ]
}
```

> **Note**
>
> Dalam versi kebijakan 2012-10-17, variabel kebijakan dimulai dengan `$`. Perubahan sintaks ini berpotensi membuat konflik jika kunci objek Anda (nama objek) termasuk `$`. Untuk menghindari konflik ini, tentukan `$` karakter dengan menggunakan`${$}`. Misalnya, untuk menyertakan kunci objek `my$file` dalam kebijakan, tentukan sebagai`my${$}file`.

Meskipun nama pengguna IAM adalah pengidentifikasi yang dapat dibaca manusia dan mudah dikenali, tetapi nama-nama tersebut tidak harus bersifat unik secara global. Misalnya, jika pengguna Carlos meninggalkan organisasi dan Carlos lainnya bergabung, maka Carlos baru dapat mengakses informasi Carlos lama.

Alih-alih menggunakan nama pengguna, Anda dapat membuat folder berdasarkan ID pengguna IAM. Setiap ID pengguna IAM adalah unik. Dalam hal ini, Anda harus mengubah kebijakan sebelumnya untuk menggunakan variabel kebijakan`${aws:userid}`. Untuk informasi lebih lanjut tentang pengidentifikasi pengguna, lihat Pengidentifikasi IAM dalam Panduan Pengguna IAM.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::example-s3-bucket1/home/${aws:userid}/*"
      }
   ]
}
```

Mengizinkan pengguna non-IAM (pengguna aplikasi seluler) mengakses folder di dalam bucket

Misalkan Anda ingin mengembangkan sebuah aplikasi seluler, sebuah permainan yang menyimpan data pengguna dalam bucket S3. Untuk setiap pengguna aplikasi, Anda ingin membuat folder dalam bucket Anda. Anda juga ingin membatasi akses setiap pengguna ke folder mereka sendiri.

Tetapi Anda tidak dapat membuat folder sebelum seseorang mengunduh aplikasi Anda dan mulai memainkan game, karena Anda tidak memiliki ID pengguna mereka.

Dalam hal ini, Anda dapat meminta pengguna untuk masuk ke aplikasi Anda dengan menggunakan penyedia identitas publik seperti Login dengan Amazon, Facebook, atau Google. Setelah pengguna masuk ke aplikasi Anda melalui salah satu penyedia layanan ini, mereka memiliki ID pengguna yang dapat Anda gunakan untuk membuat folder kustom pengguna saat runtime.

Anda kemudian dapat menggunakan federasi identitas web AWS Security Token Service untuk mengintegrasikan informasi dari penyedia identitas dengan aplikasi Anda dan untuk mendapatkan kredensyal keamanan sementara untuk setiap pengguna. Anda kemudian dapat membuat kebijakan IAM yang memungkinkan aplikasi mengakses bucket Anda dan melakukan operasi seperti membuat folder kustom pengguna dan mengunggah data. Untuk informasi lebih lanjut tentang federasi identitas web, lihat Tentang Federasi Identitas Web dalam Panduan Pengguna IAM.

Mengizinkan grup memiliki folder bersama di Amazon S3

Melampirkan kebijakan berikut ke grup akan memberikan kepada semua orang di grup akses ke folder berikut di Amazon S3: `example-s3-bucket1/share/marketing`. Anggota grup diizinkan untuk hanya mengakses izin Amazon S3 tertentu saja yang ditampilkan dalam kebijakan dan hanya untuk objek di folder tertentu saja.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion"
         ],
         "Resource":"arn:aws:s3:::example-s3-bucket1/share/marketing/*"
      }
   ]
}
```

Mengizinkan semua pengguna Anda membaca objek di sebagian bucket

Dalam contoh ini, Anda membuat grup dengan nama *AllUsers*, yang berisi semua pengguna IAM yang dimiliki oleh Akun AWS. Anda kemudian melampirkan kebijakan yang memberikan grup akses ke `GetObject` dan `GetObjectVersion`, tetapi hanya untuk objek yang ada di folder *example-s3-bucket1*`/readonly` saja.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion"
         ],
         "Resource":"arn:aws:s3:::example-s3-bucket1/readonly/*"
      }
   ]
}
```

Mengizinkan mitra untuk menempatkan file ke dalam bagian tertentu bucket

Dalam contoh ini, Anda membuat grup yang disebut *AnyCompany* yang mewakili perusahaan mitra. Anda membuat pengguna IAM untuk orang atau aplikasi tertentu di perusahaan mitra yang memerlukan akses, lalu Anda menempatkan pengguna tersebut dalam grup.

Anda kemudian melampirkan kebijakan yang memberikan `PutObject` akses kepada grup ke folder berikut dalam bucket:

example-s3-bucket1`/uploads/anycompany`

Anda ingin mencegah *AnyCompany* grup melakukan hal lain dengan bucket, sehingga Anda menambahkan pernyataan yang secara jelas menolak izin untuk tindakan Amazon S3 kecuali pada sumber daya `PutObject` Amazon S3 di. Akun AWS

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
```

```
            "Action":"s3:PutObject",
            "Resource":"arn:aws:s3:::example-s3-bucket1/uploads/anycompany/*"
        },
        {
            "Effect":"Deny",
            "Action":"s3:*",
            "NotResource":"arn:aws:s3:::example-s3-bucket1/uploads/anycompany/*"
        }
    ]
}
```

Membatasi akses ke bucket Amazon S3 dalam bucket tertentu Akun AWS

Jika Anda ingin memastikan bahwa kepala sekolah Amazon S3 Anda hanya mengakses sumber daya yang ada di dalam tepercaya Akun AWS, Anda dapat membatasi akses. Misalnya, kebijakan IAM berbasis identitas ini menggunakan `Deny` efek untuk memblokir akses ke tindakan Amazon S3, kecuali sumber daya Amazon S3 yang sedang diakses ada di akun *222222222222*. Untuk mencegah prinsipal IAM dalam mengakses objek Amazon S3 di luar akun, lampirkan kebijakan IAM berikut: Akun AWS

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyS3AccessOutsideMyBoundary",
      "Effect": "Deny",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "222222222222"
          ]
        }
      }
    }
  ]
}
```

> **Note**
>
> Kebijakan ini tidak menggantikan kontrol akses IAM yang ada, karena tidak memberikan akses apa pun. Sebagai gantinya, kebijakan ini bertindak sebagai pagar pembatas tambahan untuk izin IAM Anda yang lain, terlepas dari izin yang diberikan melalui kebijakan IAM lainnya.

Pastikan untuk mengganti ID akun *222222222222* di polis dengan ID Anda sendiri Akun AWS. Untuk menerapkan kebijakan ke beberapa akun sambil tetap mempertahankan batasan ini, ganti ID akun dengan kunci `aws:PrincipalAccount` kondisi. Kondisi ini mensyaratkan bahwa pengguna utama dan sumber daya harus berada dalam akun yang sama.

Membatasi akses ke bucket Amazon S3 dalam unit organisasi Anda

Jika Anda memiliki unit organisasi (OU) yang disiapkan AWS Organizations, Anda mungkin ingin membatasi akses bucket Amazon S3 ke bagian tertentu dari organisasi Anda. Dalam contoh ini, kami akan menggunakan `aws:ResourceOrgPaths` kunci untuk membatasi akses bucket Amazon S3 ke OU di organisasi Anda. Untuk contoh ini, ID OU adalah*ou-acroot-exampleou*. Pastikan untuk mengganti nilai ini dalam kebijakan Anda sendiri dengan ID OU Anda sendiri.

```
{
   "Version": "2012-10-17",
   "Statement": [
     {
       "Sid": "AllowS3AccessOutsideMyBoundary",
       "Effect": "Allow",
       "Action": [
         "s3:*"
       ],
       "Resource": "*",
       "Condition": {
         "ForAllValues:StringNotLike": {
           "aws:ResourceOrgPaths": [
             "o-acorg/r-acroot/ou-acroot-exampleou/"
           ]
         }
       }
     }
   ]
 }
```

> **Note**
>
> Kebijakan ini tidak memberikan akses apa pun. Sebagai gantinya, kebijakan ini bertindak sebagai penghalang untuk izin IAM Anda yang lain, mencegah pengguna utama mengakses objek Amazon S3 di luar batas yang ditentukan OU.

Kebijakan ini menolak akses ke tindakan Amazon S3 kecuali objek Amazon S3 yang sedang diakses ada di OU di *`ou-acroot-exampleou`* organisasi Anda. Kondisi kebijakan IAM mengharuskan`aws:ResourceOrgPaths`, kunci kondisi multivalued, untuk memuat salah satu jalur OU yang terdaftar. Kebijakan ini menggunakan `ForAllValues:StringNotLike` operator untuk membandingkan nilai OU yang terdaftar tanpa `aws:ResourceOrgPaths` pencocokan peka huruf besar/kecil.

Membatasi akses ke bucket Amazon S3 di organisasi Anda

Untuk membatasi akses ke objek Amazon S3 dalam organisasi Anda, lampirkan kebijakan IAM ke root organisasi, terapkan ke semua akun di organisasi Anda. Untuk mewajibkan pengguna utama IAM Anda mengikuti aturan ini, gunakan kebijakan kontrol layanan (SCP). Jika Anda memilih untuk menggunakan SCP, pastikan untuk menguji SCP secara menyeluruh sebelum melampirkan kebijakan ke root organisasi.

Dalam contoh kebijakan berikut, akses ditolak ke tindakan Amazon S3 kecuali objek Amazon S3 yang diakses berada di organisasi yang sama dengan pengguna utama IAM yang mengaksesnya:

```
{
   "Version": "2012-10-17",
   "Statement": [
     {
       "Sid": "DenyS3AccessOutsideMyBoundary",
       "Effect": "Deny",
       "Action": [
         "s3:*"
       ],
       "Resource": "arn:aws:s3:::*/*",
       "Condition": {
         "StringNotEquals": {
           "aws:ResourceOrgID": "${aws:PrincipalOrgID}"
         }
       }
```

```
      }
    ]
  }
```

> **Note**
>
> Kebijakan ini tidak memberikan akses apa pun. Sebagai gantinya, kebijakan ini bertindak sebagai penghalang untuk izin IAM Anda yang lain, mencegah pengguna utama mengakses objek Amazon S3 apa pun di luar organisasi Anda. Kebijakan ini juga berlaku untuk sumber daya Amazon S3 yang dibuat setelah kebijakan diberlakukan.

Kondisi kebijakan IAM dalam contoh ini mengharuskan `aws:ResourceOrgID` dan `aws:PrincipalOrgID` agar sama satu dengan yang lainnya. Dengan persyaratan ini, pengguna utama yang membuat permintaan dan sumber daya yang diakses harus berada di organisasi yang sama.

Memberikan izin untuk mengambil PublicAccessBlock konfigurasi untuk Akun AWS

Contoh kebijakan berbasis identitas berikut memberikan `s3:GetAccountPublicAccessBlock` izin kepada pengguna. Untuk izin ini, Anda mengatur nilai `Resource` ke `"*"`. Untuk informasi tentang ARN sumber daya, lihatSumber daya kebijakan untuk Amazon S3.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Action":[
            "s3:GetAccountPublicAccessBlock"
         ],
         "Resource":[
            "*"
         ]
       }
    ]
}
```

Membatasi pembuatan bucket ke satu Wilayah

Misalkan Akun AWS administrator ingin memberikan izin kepada penggunanya (Dave) untuk membuat ember di Wilayah Amerika Selatan (São Paulo) saja. Administrator akun dapat melampirkan kebijakan pengguna berikut ini yang memberikan izin `s3:CreateBucket` dengan syarat seperti yang ditunjukkan. Pasangan nilai kunci dalam pemblokiran `Condition` menentukan kunci `s3:LocationConstraint` dan Wilayah `sa-east-1` sebagai nilainya.

> **Note**
>
> Dalam contoh ini, pemilik bucket memberikan izin kepada salah satu penggunanya, sehingga kebijakan bucket atau kebijakan pengguna dapat digunakan. Contoh ini menunjukkan kebijakan pengguna.

Untuk daftar Wilayah Amazon S3, lihat Wilayah dan Titik Akhir di Referensi Umum AWS.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Action": "s3:CreateBucket",
         "Resource": "arn:aws:s3:::*",
         "Condition": {
             "StringLike": {
                 "s3:LocationConstraint": "sa-east-1"
             }
         }
       }
    ]
}
```

Menambahkan penolakan eksplisit

Kebijakan sebelumnya membatasi pengguna untuk membuat bucket di Wilayah lain kecuali `sa-east-1`. Namun, beberapa kebijakan lain mungkin memberikan izin kepada pengguna ini untuk membuat bucket di Wilayah lain. Misalnya, jika pengguna merupakan bagian dari sebuah kelompok, dan kelompok tersebut mungkin memiliki kebijakan yang melekat padanya yang memberikan izin kepada semua pengguna dalam kelompok tersebut untuk membuat bucket di Wilayah lain. Untuk

memastikan bahwa pengguna tidak mendapatkan izin untuk membuat bucket di Wilayah lain, Anda dapat menambahkan pernyataan penolakan eksplisit dalam kebijakan di atas.

Pernyataan `Deny` menggunakan kondisi `StringNotLike`. Artinya, permintaan buat bucket ditolak jika kendala lokasi tidak `sa-east-1`. Penolakan eksplisit tidak memungkinkan pengguna membuat bucket di Wilayah lain mana pun, apa pun izin lain yang didapat pengguna. Kebijakan berikut mencakup pernyataan penolakan eksplisit.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Action": "s3:CreateBucket",
         "Resource": "arn:aws:s3:::*",
         "Condition": {
             "StringLike": {
                 "s3:LocationConstraint": "sa-east-1"
             }
         }
       },
      {
         "Sid":"statement2",
         "Effect":"Deny",
         "Action": "s3:CreateBucket",
         "Resource": "arn:aws:s3:::*",
         "Condition": {
             "StringNotLike": {
                 "s3:LocationConstraint": "sa-east-1"
             }
         }
       }
    ]
}
```

Uji kebijakan dengan AWS CLI

Anda dapat menguji kebijakan menggunakan `create-bucket` AWS CLI perintah berikut. Contoh ini menggunakan file `bucketconfig.txt` untuk menentukan batasan lokasi. Perhatikan jalur Windows file. Anda perlu memperbarui nama bucket dan alurnya sesuai kebutuhan. Anda harus memberikan kredensial pengguna dengan menggunakan parameter `--profile`. Untuk informasi

selengkapnya tentang pengaturan dan penggunaan AWS CLI, lihatMengembangkan dengan Amazon S3 menggunakan AWS CLI.

```
aws s3api create-bucket --bucket examplebucket --profile AccountADave --create-bucket-
configuration file://c:/Users/someUser/bucketconfig.txt
```

File `bucketconfig.txt` menentukan konfigurasi sebagai berikut.

```
{"LocationConstraint": "sa-east-1"}
```

Mengontrol akses ke bucket dengan kebijakan pengguna

Panduan ini menjelaskan cara kerja izin pengguna dengan Amazon S3. Dalam contoh ini, Anda membuat bucket dengan folder. Anda kemudian membuat pengguna AWS Identity and Access Management IAM di Anda Akun AWS dan memberikan izin tambahan kepada pengguna tersebut di bucket Amazon S3 Anda dan folder di dalamnya.

Topik

- Basic bucket dan folder
- Ringkasan panduan
- Bersiap untuk panduan
- Langkah 1: Buat bucket
- Langkah 2: Buat pengguna dan grup IAM
- Langkah 3: Verifikasi bahwa pengguna IAM tidak memiliki izin
- Langkah 4: Berikan izin tingkat grup
- Langkah 5: Berikan izin tertentu kepada Alice pengguna IAM
- Langkah 6: Berikan izin tertentu kepada Bob pengguna IAM
- Langkah 7: Amankan folder privat
- Langkah 8: Membersihkan
- Sumber daya terkait

Basic bucket dan folder

Model data Amazon S3 bersifat struktur yang datar: Anda membuat bucket, dan bucket menyimpan objek tersebut. Tidak ada hierarki sub-bucket atau sub-folder, tetapi Anda dapat berusaha menyamai

hierarki folder. Alat-alat seperti konsol Amazon S3 dapat menyajikan tampilan folder dan sub-folder logika ini di dalam bucket.

Konsol tersebut menunjukkan bahwa bucket bernama `companybucket` memiliki tiga folder, `Private`, `Development`, dan `Finance`, dan sebuah objek, `s3-dg.pdf`. Konsol tersebut menggunakan nama objek (kunci) untuk membuat hierarki logis dengan folder dan subfolder. Pertimbangkan contoh berikut:

- Saat Anda membuat folder `Development`, konsol tersebut membuat objek dengan kunci `Development/`. Perhatikan jejak garis miring (`/`) pembatas.
- Saat Anda mengunggah objek yang diberi nama `Projects1.xls` dalam folder `Development`, konsol mengunggah objek dan memberikannya kunci `Development/Projects1.xls`.

  Dalam kunci tersebut, `Development` adalah prefiks dan `/` adalah pembatas. Amazon S3 API mendukung prefiks dan pembatas dalam operasinya. Sebagai contoh, Anda dapat memperoleh daftar semua objek dari bucket dengan prefiks dan pembatas tertentu. Pada konsol, saat Anda membuka folder `Development`, konsol akan mencantumkan objek yang ada di folder tersebut. Pada contoh berikut, folder `Development` berisi satu objek.

  Saat konsol mencantumkan folder `Development` di dalam bucket `companybucket`, konsol tersebut mengirimkan permintaan ke Amazon S3 yang menyebutkan prefiks dari `Development` dan pembatas dari `/` dalam permintaan. Respons konsol terlihat seperti daftar folder di sistem file komputer Anda. Contoh sebelumnya menunjukkan bahwa bucket `companybucket` memiliki sebuah objek dengan kunci `Development/Projects1.xls`.

Konsol menggunakan kunci objek untuk membuat hierarki logika. Amazon S3 tidak memiliki hierarki fisik. Amazon S3 hanya memiliki bucket yang berisi objek dalam struktur file datar. Saat Anda membuat objek menggunakan Amazon S3 API, Anda dapat menggunakan kunci objek yang menyiratkan hierarki logika. Saat Anda membuat hierarki logika dari objek-objek tersebut, Anda dapat mengelola akses ke setiap folder, seperti yang ditunjukkan oleh panduan ini.

Sebelum memulai, pastikan bahwa Anda sudah terbiasa dengan konsep konten bucket tingkat root. Misalkan bucket `companybucket` memiliki objek-objek berikut:

- `Private/privDoc1.txt`
- `Private/privDoc2.zip`
- `Development/project1.xls`

- `Development/project2.xls`
- `Finance/Tax2011/document1.pdf`
- `Finance/Tax2011/document2.pdf`
- `s3-dg.pdf`

Kunci-kunci objek ini membuat hierarki logika dengan `Private`, `Development`, dan sebagai folder tingkat root `Finance` dan sebagai objek tingkat root `s3-dg.pdf`. Saat Anda memilih nama bucket di konsol Amazon S3, item tingkat root muncul. Konsol menunjukkan prefiks tingkat atas (`Private/`, `Development/`, dan `Finance/`) sebagai folder tingkat root. Kunci objek `s3-dg.pdf` tidak memiliki prefiks, sehingga muncul sebagai item tingkat root.

Ringkasan panduan

Dalam panduan ini, Anda membuat sebuah bucket dengan tiga folder (`Private`, `Development`, dan `Finance`) di dalamnya.

Anda memiliki dua pengguna, Alice dan Bob. Anda ingin Alice hanya mengakses folder `Development`, dan Anda ingin Bob hanya mengakses folder `Finance`. Anda ingin membuat konten folder `Private` tetap privat. Dalam panduan, Anda mengelola akses dengan membuat pengguna IAM (contoh menggunakan nama pengguna Alice dan Bob) dan memberi mereka izin yang diperlukan.

IAM juga mendukung pembuatan grup pengguna dan pemberian izin tingkat grup yang berlaku untuk semua pengguna yang ada dalam grup. Hal ini membantu Anda untuk mengelola izin dengan lebih baik. Untuk latihan ini, baik Alice maupun Bob memerlukan beberapa izin yang sama. Jadi, Anda juga membuat kelompok bernama `Consultants`, lalu menambahkan Alice dan Bob ke dalam grup tersebut. Anda terlebih dahulu harus memberikan izin dengan melampirkan kebijakan grup ke grup tersebut. Kemudian Anda menambahkan izin kustom pengguna dengan melampirkan kebijakan kepada pengguna tertentu.

> **Note**
>
> Panduan menggunakan `companybucket` sebagai nama bucket, Alice dan Bob sebagai pengguna IAM, dan `Consultants` sebagai nama grup. Karena Amazon S3 mewajibkan bahwa nama bucket harus bersifat unik secara global, maka Anda harus mengganti nama bucket dengan nama yang Anda buat.

Bersiap untuk panduan

Dalam contoh ini, Anda menggunakan Akun AWS kredensyal Anda untuk membuat pengguna IAM. Pada awalnya, pengguna ini tidak memiliki izin. Anda kemudian memberikan pengguna tersebut izin tambahan untuk melakukan tindakan Amazon S3 tertentu. Untuk menguji izin ini, Anda masuk ke konsol dengan menggunakan kredensial masing-masing pengguna. Saat Anda secara bertahap memberikan izin sebagai Akun AWS pemilik dan menguji izin sebagai pengguna IAM, Anda harus masuk dan keluar, setiap kali menggunakan kredensyal yang berbeda. Anda dapat melakukan pengujian ini dengan satu browser, tetapi prosesnya akan berjalan lebih cepat jika Anda dapat menggunakan dua browser yang berbeda. Gunakan satu browser untuk terhubung AWS Management Console dengan Akun AWS kredensyal Anda dan browser lain untuk terhubung dengan kredensyal pengguna IAM.

Untuk masuk ke AWS Management Console dengan Akun AWS kredensyal Anda, buka https://console.aws.amazon.com/. Pengguna IAM tidak dapat masuk menggunakan tautan yang sama. Pengguna IAM harus menggunakan halaman masuk dengan IAM yang diaktifkan. Sebagai pemilik akun, Anda dapat memberikan tautan ini kepada pengguna Anda.

Untuk informasi lebih lanjut tentang IAM, lihat Halaman Masuk AWS Management Console dalam Panduan Pengguna IAM.

Untuk menyediakan tautan masuk bagi pengguna IAM

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada jendela Navigasi, pilih Dasbor IAM .
3. Perhatikan URL dalam Tautan masuk pengguna IAM:. Anda akan memberikan tautan ini ke pengguna IAM untuk masuk ke konsol dengan nama pengguna dan kata sandi IAM mereka.

Langkah 1: Buat bucket

Pada langkah ini, Anda masuk ke konsol Amazon S3 dengan Akun AWS kredensyal, membuat bucket, menambahkan folder ke bucket, dan mengunggah satu atau dua contoh dokumen di setiap folder.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Buat bucket.

Untuk step-by-step instruksi, lihatMembuat bucket.

3. Unggah satu dokumen ke bucket.

   Latihan ini mengasumsikan bahwa Anda memiliki dokumen `s3-dg.pdf` pada tingkat root bucket ini. Jika Anda mengunggah dokumen lain, ganti nama file untuk `s3-dg.pdf`.

4. Tambahkan tiga folder yang diberi nama `Private`, `Finance`, dan `Development` ke dalam bucket.

   Untuk step-by-step petunjuk cara membuat folder, lihat Mengatur objek di konsol Amazon S3 dengan menggunakan folder > di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

5. Unggah satu atau dua dokumen ke setiap folder.

   Untuk latihan ini, asumsikan bahwa Anda telah mengunggah beberapa dokumen di setiap folder, sehingga bucket tersebut memiliki objek dengan kunci-kunci berikut:

   - `Private/privDoc1.txt`
   - `Private/privDoc2.zip`
   - `Development/project1.xls`
   - `Development/project2.xls`
   - `Finance/Tax2011/document1.pdf`
   - `Finance/Tax2011/document2.pdf`
   - `s3-dg.pdf`

   Untuk step-by-step instruksi, lihatMengunggah Objek.

Langkah 2: Buat pengguna dan grup IAM

Sekarang gunakan Konsol IAM untuk menambahkan dua pengguna IAM, Alice dan Bob, ke pengguna Anda. Akun AWS Untuk step-by-step petunjuk, lihat Membuat pengguna IAM di Panduan Pengguna IAM Anda Akun AWS.

Juga buat grup administratif bernama`Consultants`. Kemudian tambahkan kedua pengguna ke grup. Untuk step-by-step petunjuknya, lihat Membuat grup pengguna IAM.

> **Warning**
>
> Saat Anda menambahkan pengguna dan grup, jangan melampirkan kebijakan apa pun yang memberikan izin kepada pengguna-pengguna ini. Awalnya, para pengguna ini tidak memiliki izin. Dalam bagian berikut ini, Anda memberikan izin sebagai tambahan. Pertama, Anda harus memastikan bahwa Anda telah menetapkan kata sandi untuk para pengguna IAM ini. Anda menggunakan kredensial para pengguna ini untuk menguji tindakan Amazon S3 dan memverifikasi bahwa izin berfungsi sesuai harapan.

Untuk step-by-step petunjuk cara membuat pengguna IAM baru, lihat Membuat pengguna IAM di Panduan Pengguna IAM Anda Akun AWS. Saat Anda membuat pengguna untuk panduan ini, pilih akses AWS Management Console dan hapus Akses terprogram.

Untuk step-by-step petunjuk cara membuat grup administratif, lihat Membuat Pengguna dan Grup Admin IAM Pertama Anda di Panduan Pengguna IAM.

Langkah 3: Verifikasi bahwa pengguna IAM tidak memiliki izin

Jika Anda menggunakan dua browser, sekarang Anda dapat menggunakan browser kedua untuk masuk ke konsol menggunakan salah satu kredensial pengguna IAM.

1. Menggunakan tautan masuk pengguna IAM (lihat Untuk menyediakan tautan masuk bagi pengguna IAM), masuklah ke AWS Management Console menggunakan salah satu kredensial pengguna IAM.
2. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

   Verifikasi pesan konsol yang memberi tahu Anda bahwa akses ditolak.

Sekarang, Anda dapat mulai memberikan izin tambahan kepada pengguna. Pertama, Anda harus melampirkan kebijakan grup yang memberikan izin yang harus dimiliki kedua pengguna.

Langkah 4: Berikan izin tingkat grup

Anda ingin pengguna dapat melakukan hal berikut ini:

- Cantumkan semua bucket yang dimiliki oleh akun induk. Untuk melakukannya, Bob dan Alice harus memiliki izin untuk tindakan `s3:ListAllMyBuckets`.

- Cantumkan item tingkat root, folder, dan objek dalam bucket `companybucket`. Untuk melakukannya, Bob dan Alice harus memiliki izin untuk tindakan `s3:ListBucket` pada bucket `companybucket`.

Pertama, Anda harus membuat kebijakan yang memberikan izin-izin tersebut, lalu Anda harus melampirkannya ke grup `Consultants`.

Langkah 4.1: Berikan izin untuk mencantumkan semua bucket

Pada langkah ini, Anda membuat kebijakan terkelola yang memberikan izin minimum kepada para pengguna untuk memungkinkan mereka mencantumkan daftar semua bucket yang dimiliki oleh akun induk. Kemudian Anda melampirkan kebijakan tersebut ke grup `Consultants`. Saat Anda melampirkan kebijakan terkelola ke pengguna atau grup, Anda memberikan izin kepada pengguna atau grup untuk mendapatkan daftar bucket yang dimiliki oleh Akun AWS induk.

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.

   > **Note**
   > Karena Anda memberikan izin kepada pengguna, masuklah dengan menggunakan kredensial Akun AWS Anda, bukan sebagai pengguna IAM.

2. Buat kebijakan terkelola.

   a. Pada panel navigasi yang ada di sebelah kiri, pilih Kebijakan, lalu pilih Buat Kebijakan.

   b. Pilih tab JSON.

   c. Salin kebijakan akses berikut dan tempelkan ke dalam kolom teks kebijakan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGroupToSeeBucketListInTheConsole",
      "Action": ["s3:ListAllMyBuckets"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::*"]
    }
  ]
```

```
}
```

Kebijakan adalah dokumen JSON. Dalam dokumen, `Statement` adalah inline objek, yang masing-masing menjelaskan izin dengan menggunakan kumpulan pasangan nilai-nama. Kebijakan sebelumnya menjelaskan satu izin tertentu. `Action` menyebutkan jenis akses. Dalam kebijakan, `s3:ListAllMyBuckets` adalah tindakan Amazon S3 yang telah ditentukan sebelumnya. Tindakan ini mencakup operasi Amazon S3 GET Service, yang mengembalikan daftar semua bucket yang dimiliki oleh pengirim yang diautentikasi. Nilai elemen `Effect` menentukan apakah izin tertentu diperbolehkan atau ditolak.

d. Pilih Tinjau Kebijakan. Pada halaman berikutnya, masukkan `AllowGroupToSeeBucketListInTheConsole` pada kolom Nama, dan kemudian pilih Buat kebijakan.

> **Note**
>
> Entri Ringkasan menampilkan pesan yang menyatakan bahwa kebijakan tidak memberikan izin apa pun. Untuk panduan ini, Anda dapat mengabaikan pesan ini dengan aman.

3. Lampirkan kebijakan terkelola `AllowGroupToSeeBucketListInTheConsole` yang Anda buat ke grup `Consultants`.

   Untuk step-by-step petunjuk untuk melampirkan kebijakan terkelola, lihat Menambahkan dan menghapus izin identitas IAM di Panduan Pengguna IAM.

   Anda melampirkan dokumen kebijakan ke pengguna dan grup IAM di konsol IAM. Karena Anda ingin kedua pengguna dapat mencantumkan bucket, Anda harus melampirkan kebijakan ke grup.

4. Uji izin.

   a. Menggunakan tautan masuk pengguna IAM (lihat Untuk menyediakan tautan masuk bagi pengguna IAM), masuklah ke konsol dengan menggunakan salah satu kredensial pengguna IAM.

   b. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

      Konsol tersebut harus mencantumkan semua bucket tetapi bukan objek yang ada dalam bucket mana pun.

Langkah 4.2: Mengaktifkan pengguna untuk mencantumkan konten tingkat root dalam bucket

Selanjutnya, Anda mengizinkan semua pengguna di grup `Consultants` untuk mencantumkan item bucket `companybucket` tingkat root. Ketika pengguna memilih bucket perusahaan pada konsol Amazon S3, pengguna dapat melihat item tingkat root yang ada dalam bucket.

> **Note**
>
> Contoh ini menggunakan `companybucket` sebagai ilustrasi. Anda harus menggunakan nama bucket yang Anda buat.

Untuk memahami permintaan yang dikirim konsol ke Amazon S3 saat Anda memilih nama bucket, respons yang dikembalikan Amazon S3, dan bagaimana konsol menafsirkan respons, periksa alirannya sedikit lebih dekat.

Saat Anda memilih nama bucket, konsol mengirimkan permintaan GET Bucket (List Objects) ke Amazon S3. Permintaan ini mencakup parameter-parameter berikut ini:

- Parameter `prefix` dengan string kosong sebagai nilainya.
- Parameter `delimiter` dengan `/` sebagai nilainya.

Berikut ini adalah contoh permintaan.

```
GET ?prefix=&delimiter=/ HTTP/1.1
Host: companybucket.s3.amazonaws.com
Date: Wed, 01 Aug  2012 12:00:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:xQE0diMbLRepdf3YB+FIEXAMPLE=
```

Amazon S3 mengembalikan respons yang menyertakan elemen `<ListBucketResult/>` berikut ini.

```
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>companybucket</Name>
  <Prefix></Prefix>
  <Delimiter>/</Delimiter>
   ...
  <Contents>
    <Key>s3-dg.pdf</Key>
    ...
```

```
  </Contents>
  <CommonPrefixes>
    <Prefix>Development/</Prefix>
  </CommonPrefixes>
  <CommonPrefixes>
    <Prefix>Finance/</Prefix>
  </CommonPrefixes>
  <CommonPrefixes>
    <Prefix>Private/</Prefix>
  </CommonPrefixes>
</ListBucketResult>
```

Kuncinya `s3-dg.pdf` objek tidak berisi pembatas garis miring (`/`), dan Amazon S3 mengembalikan kunci dalam elemen `<Contents>`. Namun, semua kunci lain dalam contoh bucket berisi pembatas `/`. Amazon S3 mengelompokkan kunci-kunci ini dan mengembalikan elemen `<CommonPrefixes>` untuk setiap nilai prefiks yang berbeda `Development/`, `Finance/`, dan `Private/` yakni substring dari awal kunci-kunci tersebut hingga peristiwa pertama pembatas `/` yang ditentukan.

Konsol menafsirkan hasil ini dan menampilkan item tingkat root sebagai tiga folder dan satu kunci objek.

Jika Bob atau Alice membuka folder Pengembangan, konsol akan mengirim permintaan GET Bucket (List Objects) ke Amazon S3 dengan parameter `prefix` dan `delimiter` yang ditetapkan dengan nilai-nilai berikut:

- Parameter `prefix` dengan nilai `Development/`.
- Parameter `delimiter` dengan nilai "`/`".

Sebagai respons, Amazon S3 mengembalikan kunci-kunci objek yang dimulai dengan prefiks yang ditentukan.

```
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>companybucket</Name>
  <Prefix>Development</Prefix>
  <Delimiter>/</Delimiter>
   ...
  <Contents>
    <Key>Project1.xls</Key>
    ...
  </Contents>
  <Contents>
```

```
    <Key>Project2.xls</Key>
    ...
  </Contents>
</ListBucketResult>
```

Konsol menampilkan kunci-kunci objek.

Sekarang, kembali ke pemberian izin pengguna untuk mencantumkan item bucket tingkat root. Untuk mencantumkan konten bucket, pengguna memerlukan izin untuk memanggil tindakan `s3:ListBucket`, sebagaimana yang diperlihatkan dalam pernyataan kebijakan berikut. Untuk memastikan bahwa mereka hanya melihat konten tingkat root, Anda harus menambahkan kondisi yang harus ditetapkan pengguna sebagai `prefix` kosong dalam permintaan—yaitu, mereka tidak diizinkan melakukan klik dua kali pada folder tingkat root. Terakhir, Anda menambahkan kondisi untuk mengharuskan akses gaya folder dengan mengharuskan permintaan pengguna untuk menyertakan parameter `delimiter` dengan nilai "/".

```
{
  "Sid": "AllowRootLevelListingOfCompanyBucket",
  "Action": ["s3:ListBucket"],
  "Effect": "Allow",
  "Resource": ["arn:aws:s3:::companybucket"],
  "Condition":{
         "StringEquals":{
             "s3:prefix":[""], "s3:delimiter":["/"]
                        }
              }
}
```

Saat Anda memilih bucket di konsol Amazon S3, konsol akan mengirimkan permintaan lokasi GET Bucket terlebih dahulu untuk menemukan Wilayah AWS tempat bucket digunakan. Kemudian konsol tersebut akan menggunakan titik akhir spesifik Wilayah untuk bucket untuk mengirimkan permintaan GET Bucket (List Objects). Akibatnya, jika pengguna akan menggunakan konsol, maka Anda harus memberikan izin untuk tindakan `s3:GetBucketLocation` seperti yang ditunjukkan dalam pernyataan kebijakan berikut.

```
{
   "Sid": "RequiredByS3Console",
   "Action": ["s3:GetBucketLocation"],
   "Effect": "Allow",
   "Resource": ["arn:aws:s3:::*"]
```

```
}
```

Untuk memungkinkan pengguna mencantumkan konten bucket tingkat root

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.

   Gunakan Akun AWS kredensyal Anda, bukan kredensyal pengguna IAM, untuk masuk ke konsol.

2. Ganti kebijakan terkelola `AllowGroupToSeeBucketListInTheConsole` yang sudah ada yang terlampir pada grup `Consultants` dengan kebijakan berikut, yang juga memungkinkan tindakan `s3:ListBucket`. Ingatlah untuk mengganti *`companybucket`*dalam kebijakan `Resource` dengan nama ember Anda.

   Untuk step-by-step petunjuk, lihat Mengedit kebijakan IAM di Panduan Pengguna IAM. Saat mengikuti step-by-step instruksi, pastikan untuk mengikuti langkah-langkah untuk menerapkan perubahan Anda ke semua entitas utama yang dilampirkan kebijakan tersebut.

```
{
  "Version": "2012-10-17",
  "Statement": [
     {
       "Sid":
 "AllowGroupToSeeBucketListAndAlsoAllowGetBucketLocationRequiredForListBucket",
       "Action": [ "s3:ListAllMyBuckets", "s3:GetBucketLocation" ],
       "Effect": "Allow",
       "Resource": [ "arn:aws:s3:::*"  ]
     },
     {
       "Sid": "AllowRootLevelListingOfCompanyBucket",
       "Action": ["s3:ListBucket"],
       "Effect": "Allow",
       "Resource": ["arn:aws:s3:::companybucket"],
       "Condition":{
             "StringEquals":{
                    "s3:prefix":[""], "s3:delimiter":["/"]
                           }
                   }
     }
  ]
}
```

3. Uji izin yang diperbarui.

   a. Menggunakan tautan masuk pengguna IAM (lihat Untuk menyediakan tautan masuk bagi pengguna IAM), masuklah ke AWS Management Console.

      Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

   b. Pilih bucket yang Anda buat, dan konsol akan menunjukkan item bucket tingkat root. Jika Anda memilih folder apa pun yang ada dalam bucket, maka Anda tidak akan dapat melihat konten folder tersebut karena Anda belum memberikan izin.

Pengujian ini berhasil ketika pengguna menggunakan konsol Amazon S3. Saat Anda memilih bucket pada konsol tersebut, implementasi konsol mengirim permintaan yang menyertakan parameter `prefix` dengan string kosong sebagai nilainya dan parameter `delimiter` dengan "`/`" sebagai nilainya.

Langkah 4.3: Ringkasan kebijakan grup

Hasil akhir dari kebijakan grup yang Anda tambahkan adalah memberikan kepada pengguna IAM Alice dan Bob paling tidak izin-izin berikut ini:

- Cantumkan semua bucket yang dimiliki oleh akun induk.
- Lihat item tingkat root di bucket `companybucket`.

Namun, pengguna masih tidak bisa melakukan banyak hal. Selanjutnya, Anda memberikan izin spesifik pengguna, sebagai berikut:

- Memungkinkan Alice untuk mendapatkan dan meletakkan objek di folder `Development`.
- Memungkinkan Bob untuk mendapatkan dan meletakkan objek di folder `Finance`.

Untuk izin-izin spesifik pengguna, Anda harus melampirkan kebijakan kepada pengguna tertentu, bukan ke grup. Pada bagian berikut, Anda memberikan izin kepada Alice untuk bekerja di folder `Development`. Anda dapat mengulangi langkah untuk memberikan izin serupa kepada Bob untuk bekerja di folder `Finance`.

Langkah 5: Berikan izin tertentu kepada Alice pengguna IAM

Sekarang Anda memberikan izin tambahan kepada Alice sehingga dia dapat melihat konten folder `Development` dan mengambil dan meletakkan objek di folder itu.

Langkah 5.1: Berikan izin kepada Alice pengguna IAM untuk mencantumkan konten folder pengembangan

Agar Alice dapat mencantumkan konten `Development` folder, Anda harus menerapkan kebijakan kepada pengguna Alice yang memberikan izin untuk `s3:ListBucket` tindakan di `companybucket` bucket, asalkan permintaan tersebut menyertakan awalan. `Development/` Anda ingin kebijakan ini diterapkan hanya untuk pengguna Alice, sehingga Anda menggunakan kebijakan inline. Untuk informasi selengkapnya tentang kebijakan sebaris, lihat Kebijakan terkelola dan kebijakan sebaris di Panduan Pengguna IAM.

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.

   Gunakan Akun AWS kredensyal Anda, bukan kredensyal pengguna IAM, untuk masuk ke konsol.

2. Buat kebijakan selaras untuk memberikan izin kepada pengguna Alice untuk mencantumkan konten folder `Development`.

   a. Pada panel navigasi di sebelah kiri, pilih Pengguna.

   b. Pilih nama pengguna Alice.

   c. Pada halaman rincian pengguna, pilih tab Izin dan kemudian pilih Tambahkan kebijakan selaras.

   d. Pilih tab JSON.

   e. Salin kebijakan berikut, dan tempelkan ke bidang teks kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
      "Sid": "AllowListBucketIfSpecificPrefixIsIncludedInRequest",
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::companybucket"],
      "Condition":{  "StringLike":{"s3:prefix":["Development/*"] }
       }
    }
  ]
}
```

f. Pilih Tinjau Kebijakan. Pada halaman berikutnya, masukkan nama dalam kolom Nama, dan kemudian pilih Buat kebijakan.

3. Uji perubahan izin Alice:

   a. Menggunakan tautan masuk pengguna IAM (lihat Untuk menyediakan tautan masuk bagi pengguna IAM), masuklah ke AWS Management Console.

   b. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

   c. Pada konsol Amazon S3, verifikasi bahwa Alice dapat melihat daftar objek di folder `Development/` di dalam bucket.

      Ketika pengguna memilih folder `/Development` untuk melihat daftar objek di dalamnya, konsol Amazon S3 mengirimkan permintaan `ListObjects` ke Amazon S3 dengan prefiks `/Development`. Karena pengguna diberikan izin untuk melihat daftar objek dengan prefiks `Development` dan pembatas `/`, Amazon S3 menampilkan daftar objek dengan key prefiks kunci `Development/`, dan konsol tersebut menampilkan daftar.

Langkah 5.2: Berikan izin kepada Alice pengguna IAM untuk mendapatkan dan menempatkan objek dalam folder pengembangan

Agar Alice mendapatkan dan meletakkan objek di folder `Development`, ia memerlukan izin untuk memanggil tindakan `s3:GetObject` dan `s3:PutObject`. Pernyataan kebijakan berikut memberikan izin-izin ini, dengan kondisi bahwa permintaan menyertakan parameter `prefix` dengan nilai `Development/`.

```
{
    "Sid":"AllowUserToReadWriteObjectData",
    "Action":["s3:GetObject", "s3:PutObject"],
    "Effect":"Allow",
    "Resource":["arn:aws:s3:::companybucket/Development/*"]
 }
```

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

   Gunakan Akun AWS kredensyal Anda, bukan kredensyal pengguna IAM, untuk masuk ke konsol.

2. Edit kebijakan selaras yang Anda buat pada langkah sebelumnya.

a. Pada panel navigasi di sebelah kiri, pilih Pengguna.

b. Pilih nama pengguna Alice.

c. Pada halaman rincian pengguna, pilih tab Izin dan memperluas bagian Kebijakan Selaras.

d. Di sebelah nama kebijakan yang Anda buat di langkah sebelumnya, pilih Edit Kebijakan.

e. Salin kebijakan berikut ini, dan tempelkan ke dalam kolom teks kebijakan, yang akan menggantikan kebijakan yang sudah ada.

```
{
     "Version": "2012-10-17",
     "Statement":[
      {
         "Sid":"AllowListBucketIfSpecificPrefixIsIncludedInRequest",
         "Action":["s3:ListBucket"],
         "Effect":"Allow",
         "Resource":["arn:aws:s3:::companybucket"],
         "Condition":{
            "StringLike":{"s3:prefix":["Development/*"]
            }
         }
      },
      {
        "Sid":"AllowUserToReadWriteObjectDataInDevelopmentFolder",
        "Action":["s3:GetObject", "s3:PutObject"],
        "Effect":"Allow",
        "Resource":["arn:aws:s3:::companybucket/Development/*"]
      }
   ]
}
```

3. Uji kebijakan yang diperbarui:

a. Menggunakan tautan masuk pengguna IAM (lihat Untuk menyediakan tautan masuk bagi pengguna IAM), masuklah ke AWS Management Console.

b. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

c. Pada konsol Amazon S3, verifikasi bahwa Alice kini dapat menambahkan objek dan mengunduh objek di folder `Development`.

Langkah 5.3: Tolak secara eksplisit izin Alice pengguna IAM ke folder lain yang ada dalam bucket

Pengguna Alice kini dapat mencantumkan konten tingkat root di bucket `companybucket`. Dia juga dapat mengambil dan memasukkan objek di folder `Development`. Jika Anda benar-benar ingin mengencangkan izin akses, Anda dapat secara jelas menolak akses ke folder lain yang ada dalam bucket. Jika ada kebijakan lain (kebijakan bucket atau ACL) yang memberikan Alice akses ke folder lain dalam bucket, penolakan secara jelas ini menolak izin tersebut.

Anda dapat menambahkan pernyataan berikut ke kebijakan pengguna Alice yang mengharuskan semua permintaan yang dikirimkan Alice ke Amazon S3 harus menyertakan parameter `prefix`, yang nilainya dapat berupa `Development/*` atau string kosong.

```
{
   "Sid": "ExplicitlyDenyAnyRequestsForAllOtherFoldersExceptDevelopment",
   "Action": ["s3:ListBucket"],
   "Effect": "Deny",
   "Resource": ["arn:aws:s3:::companybucket"],
   "Condition":{  "StringNotLike": {"s3:prefix":["Development/*",""] },
                  "Null"         : {"s3:prefix":false }
    }
}
```

Ada dua ekspresi bersyarat di pemblokiran `Condition`. Hasil dari ekspresi kondisional ini dikombinasikan dengan menggunakan logis `AND`. Jika kedua syarat betul, hasil dari kondisi gabungan adalah BETUL. Karena `Effect` dalam kebijakan ini adalah `Deny`, saat `Condition` dievaluasi bernilai benar, maka pengguna tidak dapat menjalankan `Action` yang ditentukan.

- Ekspresi bersyarat `Null` memastikan bahwa permintaan dari Alice menyertakan parameter `prefix`.

  Parameter `prefix` memerlukan akses seperti folder. Jika Anda mengirim permintaan tanpa parameter `prefix`, Amazon S3 mengembalikan semua kunci objek.

  Jika permintaan menyertakan parameter `prefix` dengan nilai nol, maka ekspresi dievalusi bernilai benar, dan seluruh `Condition` dievaluasi bernilai benar. Anda harus mengizinkan string kosong sebagai nilai dari parameter `prefix`. Dari pembahasan sebelumnya, ingatlah bahwa dengan memungkinkan string nol akan memungkinkan Alice mengambil item bucket tingkat root sebagaimana yang konsol lakukan dalam pembahasan sebelumnya. Untuk informasi

selengkapnya, lihat Langkah 4.2: Mengaktifkan pengguna untuk mencantumkan konten tingkat root dalam bucket.

- Ekspresi kondisional `StringNotLike` memastikan bahwa jika nilai parameter `prefix` ditentukan dan bukan `Development/*`, maka permintaan gagal.

Ikuti langkah-langkah yang diuraikan pada bagian sebelumnya dan sekali lagi perbarui kebijakan selaras yang Anda buat untuk pengguna Alice.

Salin kebijakan berikut ini, dan tempelkan ke dalam kolom teks kebijakan, yang akan menggantikan kebijakan yang sudah ada.

```
{
   "Version": "2012-10-17",
   "Statement":[
      {
         "Sid":"AllowListBucketIfSpecificPrefixIsIncludedInRequest",
         "Action":["s3:ListBucket"],
         "Effect":"Allow",
         "Resource":["arn:aws:s3:::companybucket"],
         "Condition":{
            "StringLike":{"s3:prefix":["Development/*"]
            }
         }
      },
      {
        "Sid":"AllowUserToReadWriteObjectDataInDevelopmentFolder",
        "Action":["s3:GetObject", "s3:PutObject"],
        "Effect":"Allow",
        "Resource":["arn:aws:s3:::companybucket/Development/*"]
      },
      {
         "Sid": "ExplicitlyDenyAnyRequestsForAllOtherFoldersExceptDevelopment",
         "Action": ["s3:ListBucket"],
         "Effect": "Deny",
         "Resource": ["arn:aws:s3:::companybucket"],
         "Condition":{  "StringNotLike": {"s3:prefix":["Development/*",""] },
                        "Null"         : {"s3:prefix":false }
          }
      }
   ]
}
```

Langkah 6: Berikan izin tertentu kepada Bob pengguna IAM

Sekarang Anda ingin memberikan Bob izin ke folder `Finance`. Ikuti langkah-langkah yang telah Anda gunakan sebelumnya untuk memberikan izin kepada Alice, tetapi ganti folder `Development` dengan folder `Finance`. Untuk step-by-step instruksi, lihatLangkah 5: Berikan izin tertentu kepada Alice pengguna IAM.

Langkah 7: Amankan folder privat

Dalam contoh ini, Anda hanya memiliki dua pengguna. Anda memberikan semua izin minimum yang diperlukan di tingkat grup dan memberikan izin tingkat pengguna hanya ketika Anda benar-benar perlu untuk memberikan izin pada tingkat pengguna individu. Pendekatan ini membantu meminimalkan upaya untuk mengelola izin. Saat jumlah pengguna meningkat, mengelola izin dapat menjadi rumit. Misalnya, Anda tidak ingin siapa pun dari pengguna yang ada dalam contoh ini mengakses konten folder `Private`. Bagaimana Anda memastikan bahwa Anda tidak secara tidak sengaja memberikan izin pengguna ke `Private` folder? Anda menambahkan kebijakan yang secara jelas menolak akses ke folder. Penolakan secara jelas menolak izin lainnya.

Untuk memastikan bahwa folder `Private` tetap privat, Anda dapat menambahkan dua pernyataan penolakan berikut ke kebijakan grup:

- Tambahkan pernyataan berikut untuk secara jelas menolak tindakan apa pun terhadap sumber daya dalam folder `Private` (`companybucket/Private/*`).

```
{
  "Sid": "ExplictDenyAccessToPrivateFolderToEveryoneInTheGroup",
  "Action": ["s3:*"],
  "Effect": "Deny",
  "Resource":["arn:aws:s3:::companybucket/Private/*"]
}
```

- Anda juga menolak izin untuk tindakan cantumkan objek ketika permintaan menyebutkan prefiks `Private/`. Pada konsol, jika Bob atau Alice membuka folder `Private`, maka kebijakan ini akan menyebabkan Amazon S3 mengembalikan respons kesalahan.

```
{
  "Sid": "DenyListBucketOnPrivateFolder",
  "Action": ["s3:ListBucket"],
  "Effect": "Deny",
  "Resource": ["arn:aws:s3:::*"],
  "Condition":{
```

```
      "StringLike":{"s3:prefix":["Private/"]}
   }
}
```

Mengganti kebijakan grup `Consultants` dengan kebijakan yang diperbarui yang menyertakan pernyataan penolakan sebelumnya. Setelah kebijakan yang diperbarui diterapkan, tidak ada pengguna dalam grup yang dapat mengakses folder `Private` di bucket Anda.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

   Gunakan Akun AWS kredensyal Anda, bukan kredensyal pengguna IAM, untuk masuk ke konsol.

2. Ganti kebijakan terkelola `AllowGroupToSeeBucketListInTheConsole` yang sudah ada yang terlampir pada grup `Consultants` dengan kebijakan berikut. Ingatlah untuk mengganti *`companybucket`* dalam kebijakan dengan nama bucket Anda.

   Untuk petunjuk, lihat Mengedit kebijakan yang dikelola pelanggan di Panduan Pengguna IAM. Ketika mengikuti petunjuk, pastikan untuk mengikuti petunjuk untuk menerapkan perubahan Anda ke semua entitas pengguna utama yang kepadanya kebijakan tersebut dilampirkan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid":
 "AllowGroupToSeeBucketListAndAlsoAllowGetBucketLocationRequiredForListBucket",
      "Action": ["s3:ListAllMyBuckets", "s3:GetBucketLocation"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::*"]
    },
    {
      "Sid": "AllowRootLevelListingOfCompanyBucket",
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::companybucket"],
      "Condition":{
          "StringEquals":{"s3:prefix":[""]}
       }
    },
```

```
    {
      "Sid": "RequireFolderStyleList",
      "Action": ["s3:ListBucket"],
      "Effect": "Deny",
      "Resource": ["arn:aws:s3:::*"],
      "Condition":{
          "StringNotEquals":{"s3:delimiter":"/"}
       }
     },
    {
      "Sid": "ExplictDenyAccessToPrivateFolderToEveryoneInTheGroup",
      "Action": ["s3:*"],
      "Effect": "Deny",
      "Resource":["arn:aws:s3:::companybucket/Private/*"]
    },
    {
      "Sid": "DenyListBucketOnPrivateFolder",
      "Action": ["s3:ListBucket"],
      "Effect": "Deny",
      "Resource": ["arn:aws:s3:::*"],
      "Condition":{
          "StringLike":{"s3:prefix":["Private/"]}
       }
    }
  ]
}
```

Langkah 8: Membersihkan

Untuk membersihkan, buka Konsol IAM dan hapus pengguna Alice dan Bob. Untuk step-by-step petunjuk, lihat Menghapus pengguna IAM di Panduan Pengguna IAM.

Untuk memastikan bahwa Anda tidak lagi dikenakan biaya untuk penyimpanan, Anda juga harus menghapus objek dan bucket yang Anda buat untuk latihan ini.

Sumber daya terkait

- Mengelola kebijakan IAM dalam Panduan Pengguna IAM

Panduan yang menggunakan kebijakan untuk mengelola akses ke sumber daya Amazon S3

Topik ini menyediakan contoh penelusuran pendahuluan berikut ini untuk memberikan akses ke sumber daya Amazon S3. Contoh-contoh ini menggunakan AWS Management Console untuk membuat sumber daya (bucket, objek, pengguna) dan memberi mereka izin. Contoh-contoh ini kemudian menunjukkan kepada Anda cara memverifikasi izin menggunakan alat baris perintah, sehingga Anda tidak perlu menulis kode apa pun. Kami menyediakan perintah menggunakan kedua AWS Command Line Interface (AWS CLI) dan AWS Tools for Windows PowerShell.

- Contoh 1: Pemilik bucket yang memberikan izin bucket kepada penggunanya

  Secara bawaan, pengguna IAM yang Anda buat di akun Anda tidak memiliki izin. Dalam latihan ini, Anda memberikan izin kepada pengguna untuk melakukan operasi bucket dan objek.

- Contoh 2: Pemilik bucket yang memberikan izin bucket lintas akun

  Dalam latihan ini, pemilik bucket, Akun A, memberikan izin lintas akun ke Akun AWS lain, Akun B. Akun B kemudian mendelegasikan izin tersebut kepada pengguna dalam akunnya.

- Mengelola izin objek saat pemilik objek dan bucket tidak sama

  Skenario contoh dalam kasus ini adalah tentang pemilik bucket yang memberikan izin objek kepada orang lain, tetapi tidak semua objek dalam bucket dimiliki oleh pemilik bucket. Apa izin yang diperlukan pemilik bucket, dan bagaimana izin tersebut dapat didelegasikan?

  Akun AWS Yang membuat ember disebut pemilik ember. Pemilik dapat memberikan Akun AWS izin lain untuk mengunggah objek, dan Akun AWS yang membuat objek memilikinya. Pemilik bucket tidak memiliki izin pada objek yang dibuat oleh Akun AWS lain. Jika pemilik bucket menulis kebijakan bucket yang memberikan akses ke objek, kebijakan tersebut tidak berlaku untuk objek yang dimiliki oleh akun lain.

  Dalam hal ini, pemilik objek harus terlebih dahulu memberikan izin kepada pemilik bucket dengan menggunakan ACL objek. Pemilik bucket kemudian dapat mendelegasikan izin objek tersebut kepada orang lain, kepada pengguna di akunnya sendiri, atau ke akun lain Akun AWS, seperti yang diilustrasikan oleh contoh berikut.

  - Contoh 3: Pemilik bucket yang memberikan izin kepada penggunanya ke objek yang bukan miliknya

Dalam latihan ini, pemilik bucket terlebih dahulu mendapatkan izin dari pemilik objek. Pemilik bucket kemudian mendelegasikan izin tersebut kepada pengguna dalam akunnya sendiri.

- Contoh 4 - Pemilik bucket memberikan izin lintas akun ke objek yang tidak dimilikinya

  Setelah menerima izin dari pemilik objek, pemilik bucket tidak dapat mendelegasikan izin ke yang lain Akun AWS karena delegasi lintas akun tidak didukung (lihat). Delegasi izin Sebagai gantinya, pemilik bucket dapat membuat peran IAM dengan izin untuk melakukan operasi tertentu (seperti get object) dan mengizinkan orang lain Akun AWS untuk mengambil peran tersebut. Siapa pun yang akan mengambil peran tersebut kemudian dapat mengakses objeknya. Contoh ini menunjukkan bagaimana pemilik bucket dapat menggunakan peran IAM untuk memungkinkan delegasi lintas akun ini.

Sebelum Anda mencoba contoh penelusuran

Contoh-contoh ini menggunakan AWS Management Console untuk membuat sumber daya dan memberikan izin. Untuk menguji izin, contoh menggunakan alat baris perintah, dan AWS CLI AWS Tools for Windows PowerShell, jadi Anda tidak perlu menulis kode apa pun. Untuk menguji izin, Anda harus menyiapkan salah satu alat ini. Untuk informasi selengkapnya, lihat Menyiapkan alat untuk penelusuran.

Selain itu, saat membuat sumber daya, contoh-contoh ini tidak menggunakan kredensyal pengguna root dari file. Akun AWS Sebaliknya, Anda membuat pengguna administrator di akun ini untuk melakukan tugas-tugas tersebut.

Tentang menggunakan pengguna administrator untuk membuat sumber daya dan memberikan izin

AWS Identity and Access Management (IAM) merekomendasikan untuk tidak menggunakan kredensial pengguna root perangkat Akun AWS untuk mengajukan permintaan. Sebagai gantinya, buat pengguna IAM atau peran, berikan mereka akses penuh, kemudian gunakan kredensialnya untuk mengajukan permintaan. Kami menyebut ini sebagai pengguna administratif atau peran. Untuk informasi lebih lanjut, buka Pengguna root akun AWS kredensial dan identitas IAM di Referensi Umum AWS dan Praktik Terbaik IAM di Panduan Pengguna IAM.

Semua panduan contoh dalam bagian ini menggunakan kredensial pengguna administrator. Jika Anda belum membuat pengguna administrator untuk Anda Akun AWS, topik menunjukkan caranya.

Untuk masuk ke AWS Management Console menggunakan kredensyal pengguna, Anda harus menggunakan URL Masuk pengguna IAM. Konsol IAM menyediakan URL ini untuk Anda Akun AWS. Topik-topik ini juga akan menunjukkan cara mendapatkan URL.

Menyiapkan alat untuk penelusuran

Contoh pengantar (lihatPanduan yang menggunakan kebijakan untuk mengelola akses ke sumber daya Amazon S3) menggunakan AWS Management Console untuk membuat sumber daya dan memberikan izin. Untuk menguji izin, contoh menggunakan alat baris perintah, AWS Command Line Interface (AWS CLI) dan AWS Tools for Windows PowerShell, jadi Anda tidak perlu menulis kode apa pun. Untuk menguji izin, Anda harus menyiapkan salah satu alat ini.

Untuk mengatur AWS CLI

1. Unduh dan konfigurasikan AWS CLI. Untuk instruksi, lihat topik berikut di AWS Command Line Interface Panduan Pengguna:

   Instal atau perbarui ke versi terbaru AWS Command Line Interface

   Memulai dengan AWS Command Line Interface

2. Mengatur profil bawaan.

   Anda menyimpan kredensi pengguna di file AWS CLI konfigurasi. Buat profil default di file konfigurasi menggunakan Akun AWS kredensyal Anda. Untuk petunjuk tentang menemukan dan mengedit file AWS CLI konfigurasi Anda, lihat Konfigurasi dan pengaturan file kredensyal.

   ```
   [default]
   aws_access_key_id = access key ID
   aws_secret_access_key = secret access key
   region = us-west-2
   ```

3. Verifikasi pengaturan dengan memasukkan perintah berikut pada prompt perintah. Kedua perintah ini tidak memberikan kredensial secara jelas, sehingga kredensial profil bawaan digunakan.

   - Coba `help` perintahnya.

     ```
     aws help
     ```

   - Untuk mendapatkan daftar ember pada akun yang dikonfigurasi, gunakan `aws s3 ls` perintah.

```
aws s3 ls
```

Saat Anda melalui penelusuran, Anda akan membuat pengguna, dan Anda akan menyimpan kredensyal pengguna dalam file konfigurasi dengan membuat profil, seperti yang ditunjukkan contoh berikut. Profil ini memiliki nama `AccountAadmin` dan`AccountBadmin`.

```
[profile AccountAadmin]
aws_access_key_id = User AccountAadmin access key ID
aws_secret_access_key = User AccountAadmin secret access key
region = us-west-2

[profile AccountBadmin]
aws_access_key_id = Account B access key ID
aws_secret_access_key = Account B secret access key
region = us-east-1
```

Untuk menjalankan perintah menggunakan kredensial pengguna ini, Anda perlu menambahkan parameter `--profile` yang menentukan nama profil. AWS CLI Perintah berikut mengambil daftar objek di *`examplebucket`*dan menentukan profil. `AccountBadmin`

```
aws s3 ls s3://examplebucket --profile AccountBadmin
```

Atau, Anda dapat mengonfigurasi satu set kredensial pengguna sebagai profil bawaan dengan mengubah variabel lingkungan `AWS_DEFAULT_PROFILE` dari command prompt. Setelah Anda melakukan ini, setiap kali Anda melakukan AWS CLI perintah tanpa `--profile` parameter, AWS CLI menggunakan profil yang Anda atur dalam variabel lingkungan sebagai profil default.

```
$ export AWS_DEFAULT_PROFILE=AccountAadmin
```

Untuk mengatur AWS Tools for Windows PowerShell

1. Unduh dan konfigurasikan AWS Tools for Windows PowerShell. Untuk instruksi, buka Menginstal AWS Tools for Windows PowerShell di Panduan AWS Tools for Windows PowerShell Pengguna.

> **Note**
>
> Untuk memuat AWS Tools for Windows PowerShell modul, Anda harus mengaktifkan eksekusi PowerShell skrip. Untuk informasi selengkapnya, lihat Mengaktifkan Eksekusi Skrip di Panduan AWS Tools for Windows PowerShell Pengguna.

2. Untuk penelusuran ini, Anda menentukan AWS kredensi per sesi menggunakan perintah. `Set-AWSCredentials` Perintah menyimpan kredensial ke penyimpanan persisten (parameter `-StoreAs`).

```
Set-AWSCredentials -AccessKey AccessKeyID -SecretKey SecretAccessKey -
storeas string
```

3. Verifikasi pengaturan.

   - Untuk mengambil daftar perintah yang tersedia yang dapat Anda gunakan untuk operasi Amazon S3, jalankan `Get-Command` perintah.

     ```
     Get-Command -module awspowershell -noun s3* -StoredCredentials string
     ```

   - Untuk mengambil daftar objek dalam ember, jalankan `Get-S3Object` perintah.

     ```
     Get-S3Object -BucketName bucketname -StoredCredentials string
     ```

Untuk daftar perintah, lihat AWS Alat untuk Referensi PowerShell Cmdlet.

Sekarang Anda siap untuk mencoba penelusuran. Ikuti tautan yang disediakan di awal setiap bagian.

Contoh 1: Pemilik bucket yang memberikan izin bucket kepada penggunanya

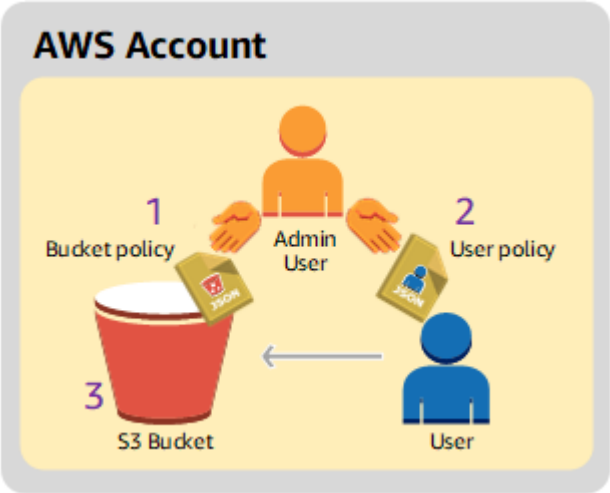
> **Important**
>
> Memberikan izin untuk peran IAM adalah praktik yang lebih baik daripada memberikan izin kepada pengguna individual.Untuk informasi selengkapnya tentang cara memberikan izin ke peran IAM, lihat. Memahami izin lintas akun dan menggunakan peran IAM

Topik

- Bersiap untuk panduan
- Langkah 1: Buat sumber daya di Akun A dan berikan izin
- Langkah 2: Menguji izin

Dalam panduan ini, a Akun AWS memiliki ember, dan akun menyertakan pengguna IAM Secara default, pengguna tidak memiliki izin. Agar pengguna dapat melakukan tugas apa pun, akun induk harus memberikan izin kepada mereka. Pemilik bucket dan akun induk sama. Oleh karena itu, untuk memberikan izin pengguna di bucket, Akun AWS dapat menggunakan kebijakan bucket, kebijakan pengguna, atau keduanya. Pemilik akun akan memberikan beberapa izin menggunakan kebijakan bucket dan beberapa izin lainnya menggunakan kebijakan pengguna.

Langkah-langkah berikut merangkum panduannya:



1. Administrator akun membuat kebijakan bucket yang memberikan serangkaian izin kepada pengguna.

2. Administrator akun melampirkan kebijakan pengguna pada pengguna yang memberikan izin tambahan.
3. Pengguna kemudian mencoba izin yang diberikan baik melalui kebijakan bucket maupun kebijakan pengguna.

Untuk contoh ini, Anda akan membutuhkan Akun AWS. Alih-alih menggunakan kredensial pengguna root akun, Anda akan membuat pengguna administrator (lihatTentang menggunakan pengguna administrator untuk membuat sumber daya dan memberikan izin). Kami merujuk ke Akun AWS dan pengguna administrator seperti yang ditunjukkan pada tabel berikut.

account-id	Akun disebut sebagai	Pengguna administrator di akun
1111-1111-1111	Akun A	AccountAadmin

> **Note**
>
> Pengguna administrator dalam contoh ini adalah AccountAadmin, yang mengacu pada Akun A, dan bukan AccountAdmin.

Semua tugas membuat pengguna dan memberikan izin dilakukan di AWS Management Console. Untuk memverifikasi izin, penelusuran menggunakan alat baris perintah, AWS Command Line Interface (AWS CLI) dan AWS Tools for Windows PowerShell, jadi Anda tidak perlu menulis kode apa pun.

Bersiap untuk panduan

1. Pastikan Anda memiliki Akun AWS dan memiliki pengguna dengan hak administrator.
   a. Mendaftar untuk Akun AWS, jika diperlukan. Kami menyebut akun ini sebagai Akun A.
      i. Buka https://aws.amazon.com/s3 dan pilih Buat AWS akun.
      ii. Ikuti petunjuk di layar.

      AWS akan memberi tahu Anda melalui email ketika akun Anda aktif dan tersedia untuk Anda gunakan.

b. Di Akun A, buat pengguna administrator**`AccountAadmin`**. Menggunakan kredensial Akun A, masuklah ke Konsol IAM dan lakukan hal-hal berikut ini:

   i. Buat pengguna **`AccountAadmin`** dan catat kredensil keamanan pengguna.

      Untuk petunjuk, lihat Membuat pengguna IAM di Panduan Pengguna IAM Anda Akun AWS.

   ii. Berikan hak administrator AccountAadmindengan melampirkan kebijakan pengguna yang memberikan akses penuh.

      Untuk instruksi, lihat Mengelola kebijakan IAM dalam Panduan Pengguna IAM.

   iii. Perhatikan URL Masuk pengguna IAM untuk. AccountAadmin Anda perlu menggunakan URL ini saat masuk ke AWS Management Console. Untuk informasi selengkapnya tentang tempat menemukan URL masuk, lihat Masuk ke AWS Management Console sebagai pengguna IAM di Panduan Pengguna IAM. Catat URL untuk setiap akun.

2. Siapkan salah satu AWS CLI atau AWS Tools for Windows PowerShell. Pastikan Anda menyimpan kredensi pengguna administrator sebagai berikut:

   - Jika menggunakan AWS CLI, buat profil,`AccountAadmin`, di file konfigurasi.
   - Jika menggunakan AWS Tools for Windows PowerShell, pastikan Anda menyimpan kredensyal untuk sesi sebagai. `AccountAadmin`

   Untuk petunjuk, lihat Menyiapkan alat untuk penelusuran.

Langkah 1: Buat sumber daya di Akun A dan berikan izin

Dengan menggunakan kredensyal pengguna `AccountAadmin` di Akun A, dan URL login pengguna IAM khusus, masuk ke akun AWS Management Console dan lakukan hal berikut:

1. Buat sumber daya bucket dan pengguna IAM

   a. Pada konsol Amazon S3, buat bucket. Perhatikan Wilayah AWS di mana Anda membuat ember. Untuk petunjuk, lihat Membuat bucket.

   b. Di Konsol IAM, lakukan hal berikut:

      i. Buat pengguna bernama Dave.

Untuk step-by-step petunjuk, lihat Membuat pengguna IAM (konsol) di Panduan Pengguna IAM.

ii. Perhatikan `UserDave` kredensialnya.

iii. Perhatikan Nama Sumber Daya Amazon (ARN) untuk pengguna Dave. Di Konsol IAM, pilih pengguna, dan tab Ringkasan menyediakan ARN pengguna.

2. Berikan izin.

Karena pemilik bucket dan akun induk yang dimiliki pengguna adalah sama, mereka Akun AWS dapat memberikan izin pengguna menggunakan kebijakan bucket, kebijakan pengguna, atau keduanya. Dalam contoh ini, Anda melakukan keduanya. Jika objek juga dimiliki oleh akun yang sama, maka pemilik bucket dapat memberikan izin objek dalam kebijakan bucket (atau kebijakan IAM).

a. Pada konsol Amazon S3, melampirkan kebijakan bucket berikut ini ke *`awsexamplebucket1`*.

Kebijakan ini memiliki dua pernyataan.

- Pernyataan pertama memberikan izin operasi bucket `s3:GetBucketLocation` dan `s3:ListBucket` kepada Dave.
- Pernyataan kedua memberikan izin kepada `s3:GetObject`. Karena Akun A juga memiliki objek, administrator akun dapat memberikan izin kepada `s3:GetObject`.

Pada pernyataan `Principal`, Dave diidentifikasi oleh pengguna ARN-nya. Untuk informasi lebih lanjut tentang elemen-elemen kebijakan, lihat Kebijakan dan izin di Amazon S3.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "statement1",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
         },
         "Action": [
            "s3:GetBucketLocation",
            "s3:ListBucket"
```

```
            ],
            "Resource": [
               "arn:aws:s3:::awsexamplebucket1"
            ]
         },
         {
            "Sid": "statement2",
            "Effect": "Allow",
            "Principal": {
               "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
            },
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
               "arn:aws:s3:::awsexamplebucket1/*"
            ]
         }
      ]
}
```

b. Buat kebijakan yang selaras untuk pengguna Dave, dengan menggunakan kebijakan berikut. Kebijakan tersebut memberikan izin `s3:PutObject` kepada Dave. Anda perlu memperbarui kebijakan dengan memberikan nama bucket Anda.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "PermissionForObjectOperations",
         "Effect": "Allow",
         "Action": [
            "s3:PutObject"
         ],
         "Resource": [
            "arn:aws:s3:::awsexamplebucket1/*"
         ]
      }
   ]
}
```

Untuk petunjuk, lihat Mengelola iAmpolicies di Panduan Pengguna IAM. Perlu diperhatikan bahwa Anda perlu masuk ke konsol dengan menggunakan kredensial Akun A.

Langkah 2: Menguji izin

Dengan menggunakan kredensial Dave, verifikasikan bahwa izin tersebut berfungsi. Anda dapat menggunakan salah satu dari dua prosedur berikut.

Izin uji menggunakan AWS CLI

1. Perbarui file AWS CLI konfigurasi dengan menambahkan `UserDaveAccountA` profil berikut. Untuk informasi selengkapnya, lihat Menyiapkan alat untuk penelusuran.

   ```
   [profile UserDaveAccountA]
   aws_access_key_id = access-key
   aws_secret_access_key = secret-access-key
   region = us-east-1
   ```

2. Verifikasi bahwa Dave dapat melakukan operasi sebagaimana yang diberikan dalam kebijakan pengguna. Unggah objek sampel menggunakan AWS CLI `put-object` perintah berikut.

   Parameter `--body` dalam perintah mengidentifikasi file sumber yang akan diunggah. Misalnya, jika file berada di root drive C: pada Windows mesin, Anda tentukan`c:\HappyFace.jpg`. Parameter `--key` menyediakan nama kunci untuk objek.

   ```
   aws s3api put-object --bucket awsexamplebucket1 --key HappyFace.jpg --
   body HappyFace.jpg --profile UserDaveAccountA
   ```

   Jalankan AWS CLI perintah berikut untuk mendapatkan objek.

   ```
   aws s3api get-object --bucket awsexamplebucket1 --key HappyFace.jpg OutputFile.jpg
    --profile UserDaveAccountA
   ```

Izin uji menggunakan AWS Tools for Windows PowerShell

1. Simpan kredensial Dave sebagai `AccountADave`. Anda kemudian menggunakan kredensyal ini ke `PUT` dan `GET` objek.

```
set-awscredentials -AccessKey AccessKeyID -SecretKey SecretAccessKey -storeas
 AccountADave
```

2. Unggah objek sampel menggunakan AWS Tools for Windows PowerShell `Write-S3Object` perintah menggunakan kredensi tersimpan pengguna Dave.

```
Write-S3Object -bucketname awsexamplebucket1 -key HappyFace.jpg -file HappyFace.jpg
 -StoredCredentials AccountADave
```

Unduh objek yang telah diunggah sebelumnya.

```
Read-S3Object -bucketname awsexamplebucket1 -key HappyFace.jpg -file Output.jpg -
StoredCredentials AccountADave
```

Contoh 2: Pemilik bucket yang memberikan izin bucket lintas akun

> **Important**
>
> Memberikan izin untuk peran IAM adalah praktik yang lebih baik daripada memberikan izin kepada pengguna individu. Untuk mempelajari cara melakukannya, lihat Memahami izin lintas akun dan menggunakan peran IAM.

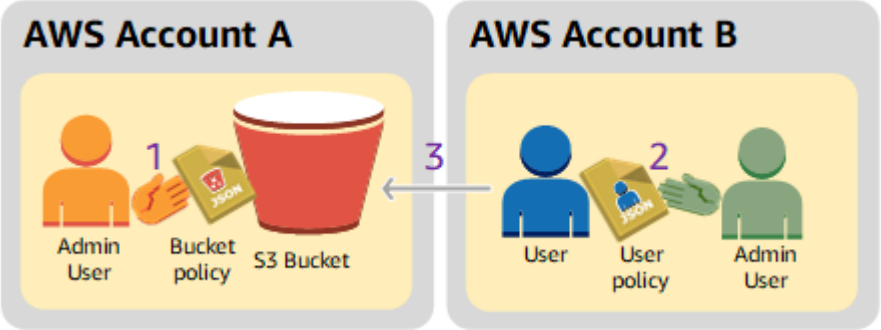
Topik

- Bersiap untuk panduan
- Langkah 1: Melakukan tugas Akun A
- Langkah 2: Melakukan tugas Akun B
- Langkah 3: (Opsional) Coba penolakan eksplisit
- Langkah 4: Membersihkan

Sebuah Akun AWS—misalnya, Akun a—dapat memberikan izin lain Akun AWS, Akun B, untuk mengakses sumber dayanya seperti ember dan objek. Akun B kemudian dapat mendelegasikan izin tersebut kepada pengguna dalam akunnya. Dalam skenario contoh ini, pemilik bucket memberikan izin lintas akun kepada akun lain untuk melakukan operasi bucket tertentu.

> **Note**
>
> Akun A juga dapat secara langsung memberikan izin kepada pengguna di Akun B dengan menggunakan kebijakan bucket. Namun, pengguna masih memerlukan izin dari akun induk, Akun B, tempat pengguna berada, bahkan jika Akun B tidak memiliki izin dari Akun A. Selama pengguna memiliki izin dari pemilik sumber daya dan akun induk, pengguna akan dapat mengakses sumber daya.

Berikut ini adalah ringkasan langkah-langkah panduan:



1. Pengguna administrator Akun A melampirkan kebijakan bucket yang memberikan izin lintas akun kepada Akun B untuk melakukan operasi bucket tertentu.

   Perhatikan bahwa pengguna administrator di Akun B akan secara otomatis mewarisi izin tersebut.
2. Pengguna administrator Akun B melampirkan kebijakan pengguna kepada pengguna yang mendelegasikan izin yang diterima pengguna administrator Akun B dari Akun A.
3. Pengguna di Akun B kemudian memverifikasi izin dengan mengakses objek yang ada dalam bucket yang dimiliki oleh Akun A.

Untuk contoh ini, Anda memerlukan dua akun. Tabel berikut menunjukkan cara kami merujuk akun-akun ini dan pengguna administrator di dalamnya. Sesuai dengan pedoman IAM (lihatTentang menggunakan pengguna administrator untuk membuat sumber daya dan memberikan izin), kami tidak menggunakan kredensyal pengguna root dalam panduan ini. Sebagai gantinya, Anda membuat pengguna administrator di setiap akun dan menggunakan kredensial tersebut saat membuat sumber daya dan memberikan mereka izin.

Akun AWS ID	Akun disebut sebagai	Pengguna administrator di akun
1111-1111-1111	Akun A	AccountAadmin
2222-2222-2222	Akun B	AccountBadmin

Semua tugas membuat pengguna dan memberikan izin dilakukan di AWS Management Console. Untuk memverifikasi izin, panduan menggunakan alat baris perintah, (AWS Command Line Interface CLI) dan AWS Tools for Windows PowerShell, jadi Anda tidak perlu menulis kode apa pun.

Bersiap untuk panduan

1. Pastikan Anda memiliki dua Akun AWS dan bahwa setiap akun memiliki satu pengguna administrator seperti yang ditunjukkan pada tabel di bagian sebelumnya.

   a. Mendaftar untuk Akun AWS, jika diperlukan.

   b. Dengan menggunakan kredensial Akun A, masuk ke Konsol IAM untuk membuat pengguna administrator:

      i. Buat pengguna **`AccountAadmin`** dan catat kredensialnya. Untuk instruksi, lihat Membuat Pengguna IAM dalam Akun AWS Anda, dalam Panduan Pengguna IAM.

      ii. Berikan hak administrator AccountAadmindengan melampirkan kebijakan pengguna yang memberikan akses penuh. Untuk instruksi, lihat Bekerja dengan Kebijakan dalam Panduan Pengguna IAM.

   c. Saat Anda berada di konsol IAM, perhatikan URL Masuk pengguna IAM di Dasbor. Semua pengguna di akun tersebut harus menggunakan URL ini saat masuk ke AWS Management Console.

      Untuk informasi lebih lanjut, lihat Cara Pengguna Masuk ke Akun Anda dalam Panduan Pengguna IAM.

   d. Ulangi langkah sebelumnya dengan menggunakan kredensial Akun B dan buat pengguna administrator **`AccountBadmin`**.

2. Siapkan AWS Command Line Interface (AWS CLI) atau AWS Tools for Windows PowerShell. Pastikan Anda menyimpan kredensi pengguna administrator sebagai berikut:

   - Jika menggunakan AWS CLI, buat dua profil, `AccountAadmin` dan`AccountBadmin`, dalam file konfigurasi.
   - Jika menggunakan AWS Tools for Windows PowerShell, pastikan bahwa Anda menyimpan kredensyal untuk sesi sebagai `AccountAadmin` dan. `AccountBadmin`

   Untuk instruksi, lihat Menyiapkan alat untuk penelusuran.

3. Simpan kredensial pengguna administrator, juga disebut sebagai profil. Anda dapat menggunakan nama profil alih-alih menentukan kredensial untuk setiap perintah yang Anda masukkan. Untuk informasi selengkapnya, lihat Menyiapkan alat untuk penelusuran.

   a. Tambahkan profil dalam file AWS CLI kredensyal untuk masing-masing pengguna administrator, `AccountAadmin` dan`AccountBadmin`, di dua akun.

```
[AccountAadmin]
aws_access_key_id = access-key-ID
aws_secret_access_key = secret-access-key
region = us-east-1

[AccountBadmin]
aws_access_key_id = access-key-ID
aws_secret_access_key = secret-access-key
region = us-east-1
```

b. Jika Anda menggunakan AWS Tools for Windows PowerShell, jalankan perintah berikut.

```
set-awscredentials -AccessKey AcctA-access-key-ID -SecretKey AcctA-secret-
access-key -storeas AccountAadmin
set-awscredentials -AccessKey AcctB-access-key-ID -SecretKey AcctB-secret-
access-key -storeas AccountBadmin
```

Langkah 1: Melakukan tugas Akun A

Langkah 1.1: Masuk ke AWS Management Console

Menggunakan URL login pengguna IAM untuk Akun A, pertama-tama masuk ke pengguna AWS Management Console sebagai AccountAadmin. Pengguna ini akan membuat bucket dan melampirkan kebijakan pada bucket tersebut.

Langkah 1.2: Buat bucket

1. Pada konsol Amazon S3, buat bucket. Latihan ini mengasumsikan ember dibuat di AS Timur (Virginia N.) Wilayah AWS dan diberi nama. *DOC-EXAMPLE-BUCKET*

   Untuk petunjuk, lihat Membuat bucket.

2. Unggah objek sampel ke bucket.

   Untuk instruksi, buka Langkah 2: Unggah satu objek ke bucket Anda.

Langkah 1.3: Melampirkan kebijakan bucket untuk memberikan izin lintas akun kepada Akun B

Kebijakan bucket memberikan `s3:ListBucket` izin `s3:GetLifecycleConfiguration` dan ke Akun B. Diasumsikan bahwa Anda masih masuk ke konsol menggunakan kredensyal AccountAadminpengguna.

1. Melampirkan kebijakan bucket berikut untuk *DOC-EXAMPLE-BUCKET*. Kebijakan tersebut memberikan izin Akun B untuk tindakan `s3:GetLifecycleConfiguration` dan `s3:ListBucket`.

   Untuk petunjuk, lihat Menambahkan kebijakan bucket dengan menggunakan konsol Amazon S3.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Example permissions",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountB-ID:root"
         },
         "Action": [
            "s3:GetLifecycleConfiguration",
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
         ]
      }
   ]
}
```

2. Verifikasi bahwa Akun B (dan dengan demikian pengguna administratornya) dapat melakukan operasi.

   - Verifikasi menggunakan AWS CLI

```
aws s3 ls s3://DOC-EXAMPLE-BUCKET --profile AccountBadmin
aws s3api get-bucket-lifecycle-configuration --bucket DOC-EXAMPLE-BUCKET --
profile AccountBadmin
```

   - Verifikasi menggunakan AWS Tools for Windows PowerShell

```
get-s3object -BucketName DOC-EXAMPLE-BUCKET -StoredCredentials AccountBadmin
get-s3bucketlifecycleconfiguration -BucketName DOC-EXAMPLE-BUCKET -
StoredCredentials AccountBadmin
```

Langkah 2: Melakukan tugas Akun B

Sekarang administrator Akun B membuat pengguna, yakni Dave, dan mendelegasikan izin yang diterima dari Akun A.

Langkah 2.1: Masuk ke AWS Management Console

Menggunakan URL login pengguna IAM untuk Akun B, pertama-tama masuk ke pengguna AWS Management Console sebagai AccountBadmin.

Langkah 2.2: Buat Dave pengguna di Akun B

Di Konsol IAM, buat pengguna,**`Dave`**.

Untuk petunjuk, lihat Membuat pengguna IAM (konsol) di Panduan Pengguna IAM.

Langkah 2.3: Delegasikan izin untuk dave pengguna

Buat kebijakan yang selaras untuk pengguna Dave, dengan menggunakan kebijakan berikut. Anda harus memperbarui kebijakan dengan memberikan nama bucket Anda.

Diasumsikan bahwa Anda masuk ke konsol menggunakan AccountBadminkredensil pengguna.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Example",
         "Effect": "Allow",
         "Action": [
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
         ]
      }
```

```
    ]
}
```

Untuk instruksi, lihat Mengelola kebijakan IAM dalam Panduan Pengguna IAM.

Langkah 2.4: Menguji izin

Sekarang Dave di Akun B dapat mencantumkan isi *DOC-EXAMPLE-BUCKET* yang dimiliki oleh Akun A. Anda dapat memverifikasi izin menggunakan salah satu prosedur berikut.

Izin uji menggunakan AWS CLI

1. Tambahkan `UserDave` profil ke file AWS CLI konfigurasi. Untuk informasi lebih lanjut tentang file config, lihat Menyiapkan alat untuk penelusuran.

   ```
   [profile UserDave]
   aws_access_key_id = access-key
   aws_secret_access_key = secret-access-key
   region = us-east-1
   ```

2. Pada command prompt, masukkan AWS CLI perintah berikut untuk memverifikasi Dave sekarang bisa mendapatkan daftar objek dari yang *DOC-EXAMPLE-BUCKET* dimiliki oleh Account A. Catatan perintah menentukan profil. `UserDave`

   ```
   aws s3 ls s3://DOC-EXAMPLE-BUCKET --profile UserDave
   ```

   Dave tidak memiliki izin lain. Jadi, jika dia mencoba operasi lain—misalnya, `get-bucket-lifecycle` konfigurasi berikut—Amazon S3 mengembalikan izin yang ditolak.

   ```
   aws s3api get-bucket-lifecycle-configuration --bucket DOC-EXAMPLE-BUCKET --profile
    UserDave
   ```

Izin uji menggunakan AWS Tools for Windows PowerShell

1. Simpan kredensial Dave sebagai `AccountBDave`.

   ```
   set-awscredentials -AccessKey AccessKeyID -SecretKey SecretAccessKey -storeas
    AccountBDave
   ```

2. Coba perintah Cantumkan bucket.

```
get-s3object -BucketName DOC-EXAMPLE-BUCKET -StoredCredentials AccountBDave
```

Dave tidak memiliki izin lain. Jadi, jika dia mencoba operasi lain—misalnya, berikut `get-s3bucketlifecycleconfiguration` ini—Amazon S3 mengembalikan izin yang ditolak.

```
get-s3bucketlifecycleconfiguration -BucketName DOC-EXAMPLE-BUCKET -
StoredCredentials AccountBDave
```

Langkah 3: (Opsional) Coba penolakan eksplisit

Anda dapat memiliki izin yang diberikan dengan menggunakan daftar kontrol akses (ACL), kebijakan bucket, atau kebijakan pengguna. Tetapi jika ada penolakan eksplisit yang ditetapkan oleh kebijakan bucket atau kebijakan pengguna, penolakan eksplisit lebih diutamakan daripada izin lainnya. Untuk pengujian, perbarui kebijakan bucket dan tolak izin Akun B secara eksplisit. `s3:ListBucket` Kebijakan tersebut juga memberikan `s3:ListBucket` izin. Namun, penolakan eksplisit diutamakan, dan Akun B atau pengguna di Akun B tidak akan dapat mencantumkan objek. *DOC-EXAMPLE-BUCKET*

1. Menggunakan kredensial pengguna `AccountAadmin` pada Akun A, ganti kebijakan kelompok dengan yang berikut ini.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Example permissions",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountB-ID:root"
         },
         "Action": [
            "s3:GetLifecycleConfiguration",
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
         ]
      },
      {
```

```
            "Sid": "Deny permission",
            "Effect": "Deny",
            "Principal": {
               "AWS": "arn:aws:iam::AccountB-ID:root"
            },
            "Action": [
               "s3:ListBucket"
            ],
            "Resource": [
               "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
            ]
         }
      ]
}
```

2. Sekarang jika Anda mencoba untuk mendapatkan daftar ember menggunakan `AccountBadmin` kredensyal, akses ditolak.
   - Menggunakan AWS CLI, jalankan perintah berikut:

     ```
     aws s3 ls s3://DOC-EXAMPLE-BUCKET --profile AccountBadmin
     ```

   - Menggunakan AWS Tools for Windows PowerShell, jalankan perintah berikut:

     ```
     get-s3object -BucketName DOC-EXAMPLE-BUCKET -StoredCredentials AccountBDave
     ```

Langkah 4: Membersihkan

1. Setelah selesai melakukan pengujian, Anda dapat melakukan hal berikut untuk membersihkan:
   - Masuk ke AWS Management Console (AWS Management Console) menggunakan kredensyal Akun A, dan lakukan hal berikut:
     - Pada konsol Amazon S3, hapus kebijakan bucket yang terlampir pada *DOC-EXAMPLE-BUCKET*. Dalam bucket Properti, hapus kebijakan yang ada dalam bagian Izin.
     - Jika bucket dibuat untuk latihan ini, pada konsol Amazon S3, hapus objek dan kemudian hapus bucket.
     - Di Konsol IAM, hapus `AccountAadmin` pengguna.
2. Masuk ke Konsol IAM menggunakan kredensil Akun B. Hapus penggunaAccountBadmin. Untuk step-by-step petunjuk, lihat Menghapus pengguna IAM di Panduan Pengguna IAM.

Contoh 3: Pemilik bucket yang memberikan izin kepada penggunanya ke objek yang bukan miliknya

> **Important**
>
> Memberikan izin untuk peran IAM adalah praktik yang lebih baik daripada memberikan izin kepada pengguna individu. Untuk mempelajari cara melakukannya, lihat Memahami izin lintas akun dan menggunakan peran IAM.

Topik

- Langkah 0: Bersiap-siap untuk panduan
- Langkah 1: Melakukan tugas Akun A
- Langkah 2: Melakukan tugas Akun B
- Langkah 3: Menguji izin
- Langkah 4: Membersihkan

Skenario untuk contoh ini adalah bahwa pemilik bucket ingin memberikan izin untuk mengakses objek, tetapi pemilik bucket tidak memiliki semua objek di bucket. Untuk contoh ini, pemilik bucket mencoba memberikan izin kepada pengguna yang ada di akunnya sendiri.

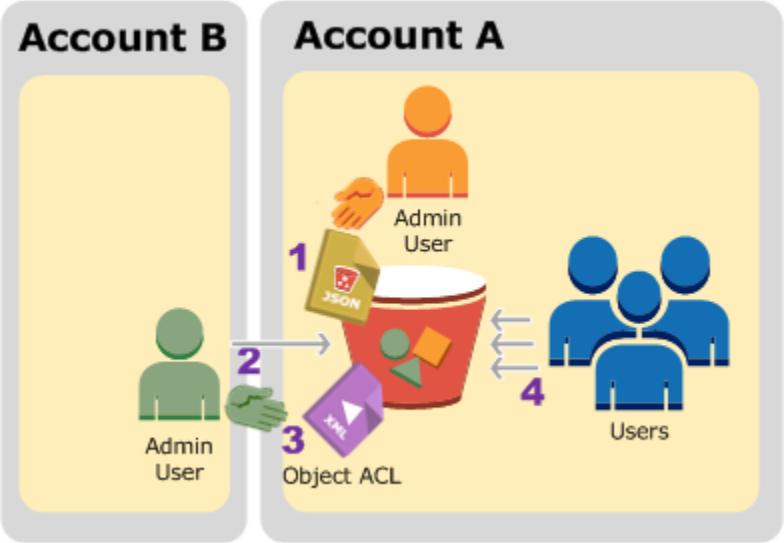
Pemilik bucket dapat mengaktifkan orang lain Akun AWS untuk mengunggah objek. Secara default, pemilik bucket tidak memiliki objek yang ditulis ke bucket oleh orang lain Akun AWS. Objek dimiliki oleh akun yang menulisnya ke bucket S3. Jika pemilik bucket tidak memiliki objek di bucket, pemilik objek harus terlebih dahulu memberikan izin kepada pemilik bucket menggunakan daftar kontrol akses objek (ACL). Kemudian, pemilik bucket dapat memberikan izin ke objek yang tidak mereka miliki. Untuk informasi selengkapnya, lihat bucket Amazon S3 dan kepemilikan objek.

Jika pemilik bucket menerapkan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan S3 Object untuk bucket, pemilik bucket akan memiliki semua objek di bucket, termasuk objek yang ditulis oleh yang lain. Akun AWS Pendekatan ini menyelesaikan masalah bahwa objek tidak dimiliki oleh pemilik ember. Kemudian, Anda dapat mendelegasikan izin kepada pengguna di akun Anda sendiri, atau ke orang lain Akun AWS.

> **Note**
>
> Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda, serta

menonaktifkan atau mengaktifkan ACL. Secara default, Kepemilikan Objek diatur ke pengaturan yang diberlakukan pemilik Bucket, dan semua ACL dinonaktifkan. Ketika ACL dinonaktifkan, pemilik bucket memiliki semua objek di bucket dan mengelola akses ke objek-objek tersebut secara eksklusif dengan menggunakan kebijakan manajemen akses. Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL. Kami menyarankan agar ACL dinonaktifkan, kecuali dalam keadaan yang tidak biasa ketika Anda perlu mengontrol akses untuk setiap objek secara individual. Dengan ACL dinonaktifkan, Anda dapat menggunakan kebijakan untuk mengontrol akses ke semua objek di bucket, terlepas dari siapa yang mengunggah objek ke bucket Anda. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Dalam contoh ini, kami menganggap pemilik bucket belum menerapkan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek. Pemilik bucket mendelegasikan izin kepada pengguna di akunnya sendiri. Berikut ini adalah ringkasan langkah-langkah penelusuran:



1. Pengguna administrator Akun A melampirkan kebijakan bucket dengan dua pernyataan.
   - Memungkinkan izin lintas akun ke Akun B untuk mengunggah objek.
   - Memungkinkan pengguna dalam akunnya sendiri untuk mengakses objek yang ada dalam bucket.
2. Pengguna administrator Akun B mengunggah objek ke bucket yang dimiliki oleh Akun A.
3. Administrator Akun B memperbarui ACL objek yang menambahkan pemberian yang memberikan pemilik bucket izin kontrol penuh pada objek.

4. Pengguna di Akun A memverifikasi dengan mengakses objek di dalam bucket, terlepas dari siapa pemiliknya.

Untuk contoh ini, Anda memerlukan dua akun. Tabel berikut menunjukkan cara kami merujuk akun-akun ini, dan pengguna administrator yang ada dalam akun-akun tersebut. Dalam panduan ini, Anda tidak menggunakan kredensial pengguna root, menurut pedoman IAM yang direkomendasikan. Untuk informasi selengkapnya, lihat Tentang menggunakan pengguna administrator untuk membuat sumber daya dan memberikan izin. Sebagai gantinya, Anda membuat administrator di setiap akun dan menggunakan kredensial tersebut untuk membuat sumber daya dan memberikan mereka izin.

Akun AWS ID	Akun disebut sebagai	Administrator pada akun
1111-1111-1111	Akun A	AccountAadmin
2222-2222-2222	Akun B	AccountBadmin

Semua tugas membuat pengguna dan memberikan izin dilakukan di AWS Management Console. Untuk memverifikasi izin, penelusuran menggunakan alat baris perintah, AWS Command Line Interface (AWS CLI) dan AWS Tools for Windows PowerShell, jadi Anda tidak perlu menulis kode apa pun.

Langkah 0: Bersiap-siap untuk panduan

1. Pastikan Anda memiliki dua Akun AWS dan setiap akun memiliki satu administrator seperti yang ditunjukkan pada tabel di bagian sebelumnya.

   a. Mendaftar untuk Akun AWS, jika diperlukan.

   b. Menggunakan kredensyal Akun A, masuk ke Konsol IAM dan lakukan hal berikut untuk membuat pengguna administrator:

      - Buat pengguna **`AccountAadmin`** dan catat kredensil keamanan pengguna. Untuk informasi lebih lanjut tentang penambahan pengguna, lihat Membuat Pengguna IAM dalam Akun AWS Anda dalam Panduan Pengguna IAM.
      - Berikan izin administrator AccountAadmindengan melampirkan kebijakan pengguna yang memberikan akses penuh. Untuk instruksi, lihat Mengelola kebijakan IAM dalam Panduan Pengguna IAM.

- Di Dasbor Konsol IAM, perhatikan URL Masuk Pengguna IAM. Semua pengguna di akun tersebut harus menggunakan URL ini saat masuk ke AWS Management Console. Untuk informasi lebih lanjut, lihat Cara pengguna masuk ke akun anda dalam Panduan Pengguna IAM.

c. Ulangi langkah sebelumnya dengan menggunakan kredensial Akun B dan buat pengguna administrator **`AccountBadmin`**.

2. Siapkan salah satu AWS CLI atau Tools untuk Windows PowerShell. Pastikan Anda menyimpan kredensial administrator seperti berikut ini:

   - Jika menggunakan AWS CLI, buat dua profil, `AccountAadmin` dan`AccountBadmin`, dalam file konfigurasi.
   - Jika menggunakan Tools untuk Windows PowerShell, pastikan bahwa Anda menyimpan kredensyal untuk sesi sebagai `AccountAadmin` dan. `AccountBadmin`

   Untuk petunjuk, lihat Menyiapkan alat untuk penelusuran.

Langkah 1: Melakukan tugas Akun A

Lakukan langkah-langkah berikut untuk Akun A:

Langkah 1.1: Masuk ke konsol tersebut

Menggunakan URL login pengguna IAM untuk Akun A, masuk ke pengguna AWS Management Console sebagai**`AccountAadmin`**. Pengguna ini akan membuat bucket dan melampirkan kebijakan pada bucket tersebut.

Langkah 1.2: Buat bucket dan pengguna, dan menambahkan kebijakan bucket yang memberikan izin pengguna

1. Pada konsol Amazon S3, buat bucket. Latihan ini mengasumsikan bahwa ember dibuat di AS Timur (Virginia N.) Wilayah AWS, dan namanya adalah. *`example-s3-bucket1`*

   Untuk petunjuk, lihat Membuat bucket.

2. Di Konsol IAM, buat pengguna**`Dave`**.

   Untuk step-by-step petunjuk, lihat Membuat pengguna IAM (konsol) di Panduan Pengguna IAM.

3. Perhatikan kredensi Dave pengguna.

4. Pada konsol Amazon S3, lampirkan kebijakan bucket berikut ini ke bucket *example-s3-bucket1*. Untuk petunjuk, lihat Menambahkan kebijakan bucket dengan menggunakan konsol Amazon S3. Ikuti langkah-langkah untuk menambahkan kebijakan bucket. Untuk informasi tentang cara menemukan ID akun, lihat Menemukan Akun AWS ID Anda.

   Kebijakan tersebut memberikan izin `s3:PutObject` dan `s3:ListBucket` kepada Akun B. Kebijakan ini juga memberikan `s3:GetObject` izin `Dave` kepada pengguna.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Statement1",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountB-ID:root"
         },
         "Action": [
            "s3:PutObject",
            "s3:ListBucket"
         ],
         "Resource": [
            "arn:aws:s3:::example-s3-bucket1/*",
            "arn:aws:s3:::example-s3-bucket1"
         ]
      },
      {
         "Sid": "Statement3",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::AccountA-ID:user/Dave"
         },
         "Action": [
            "s3:GetObject"
         ],
         "Resource": [
            "arn:aws:s3:::example-s3-bucket1/*"
         ]
      }
   ]
}
```

Langkah 2: Melakukan tugas Akun B

Setelah Akun B memiliki izin untuk melakukan operasi pada bucket Akun A, administrator Akun B melakukan hal berikut:

- Mengunggah objek ke bucket Akun A
- Menambahkan hibah di objek ACL untuk mengizinkan Akun A, pemilik bucket, kontrol penuh

Menggunakan AWS CLI

1. Menggunakan `put-object` AWS CLI perintah, unggah objek. Parameter `--body` dalam perintah mengidentifikasi file sumber yang akan diunggah. Misalnya, jika file ada di `C:` drive Windows mesin, tentukan`c:\HappyFace.jpg`. Parameter `--key` menyediakan nama kunci untuk objek.

```
aws s3api put-object --bucket example-s3-bucket1 --key HappyFace.jpg --body
 HappyFace.jpg --profile AccountBadmin
```

2. Tambahkan pemberian pada ACL objek untuk memungkinkan pemilik bucket mengontrol penuh objek. Untuk informasi tentang cara menemukan ID pengguna kanonik, lihat Menemukan ID pengguna kanonik untuk Anda Akun AWS di Panduan Referensi Manajemen AWS Akun.

```
aws s3api put-object-acl --bucket example-s3-bucket1 --key HappyFace.jpg --grant-
full-control id="AccountA-CanonicalUserID" --profile AccountBadmin
```

Menggunakan Alat untuk Windows PowerShell

1. Menggunakan `Write-S3Object` perintah, unggah objek.

```
Write-S3Object -BucketName example-s3-bucket1 -key HappyFace.jpg -file
 HappyFace.jpg -StoredCredentials AccountBadmin
```

2. Tambahkan pemberian pada ACL objek untuk memungkinkan pemilik bucket mengontrol penuh objek.

```
Set-S3ACL -BucketName example-s3-bucket1 -Key HappyFace.jpg -CannedACLName "bucket-
owner-full-control" -StoredCreden
```

Langkah 3: Menguji izin

Sekarang, verifikasi pengguna Dave di Akun A apakah dapat mengakses objek yang dimiliki oleh Akun B.

Menggunakan AWS CLI

1. Tambahkan kredensi Dave pengguna ke file AWS CLI konfigurasi dan buat profil baru,. `UserDaveAccountA` Untuk informasi selengkapnya, lihat Menyiapkan alat untuk penelusuran.

```
[profile UserDaveAccountA]
aws_access_key_id = access-key
aws_secret_access_key = secret-access-key
region = us-east-1
```

2. Jalankan perintah `get-object` CLI untuk mengunduh `HappyFace.jpg` dan menyimpannya secara lokal. Anda memberikan kredensial pengguna Dave dengan menambahkan parameter `--profile`.

```
aws s3api get-object --bucket example-s3-bucket1 --key HappyFace.jpg Outputfile.jpg
 --profile UserDaveAccountA
```

Menggunakan Alat untuk Windows PowerShell

1. Simpan AWS kredensi Dave pengguna, seperti`UserDaveAccountA`, ke toko persisten.

```
Set-AWSCredentials -AccessKey UserDave-AccessKey -SecretKey UserDave-
SecretAccessKey -storeas UserDaveAccountA
```

2. Jalankan perintah `Read-S3Object` untuk mengunduh objek `HappyFace.jpg` dan menyimpannya secara lokal. Anda memberikan kredensial ke pengguna Dave dengan menambahkan parameter `-StoredCredentials`.

```
Read-S3Object -BucketName example-s3-bucket1 -Key HappyFace.jpg -file HappyFace.jpg
  -StoredCredentials UserDaveAccountA
```

Langkah 4: Membersihkan

1. Setelah selesai melakukan pengujian, Anda dapat melakukan hal berikut untuk membersihkan:

- Masuk ke AWS Management Console menggunakan kredensial Akun A, dan lakukan hal-hal berikut ini:
  - *Di konsol Amazon S3, hapus kebijakan bucket yang dilampirkan ke example-s3-bucket1.* Dalam bucket Properti, hapus kebijakan yang ada dalam bagian Izin.
  - Jika bucket dibuat untuk latihan ini, pada konsol Amazon S3, hapus objek dan kemudian hapus bucket.
  - Di Konsol IAM, hapus AccountAadminpengguna. Untuk step-by-step petunjuk, lihat Menghapus pengguna IAM di Panduan Pengguna IAM.

2. Masuklah ke AWS Management Console menggunakan kredensial Akun B. Di Konsol IAM, hapus pengguna AccountBadmin.

Contoh 4 - Pemilik bucket memberikan izin lintas akun ke objek yang tidak dimilikinya

Topik

- Memahami izin lintas akun dan menggunakan peran IAM
- Langkah 0: Bersiap-siap untuk panduan
- Langkah 1: Melakukan tugas Akun A
- Langkah 2: Melakukan tugas Akun B
- Langkah 3: Lakukan tugas Akun C
- Langkah 4: Membersihkan
- Sumber daya terkait

Dalam skenario contoh ini, Anda memiliki bucket dan Anda telah mengaktifkan orang lain Akun AWS untuk mengunggah objek. Jika Anda telah menerapkan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan S3 Object untuk bucket, Anda akan memiliki semua objek di bucket, termasuk objek yang ditulis oleh objek lain. Akun AWS Pendekatan ini menyelesaikan masalah bahwa objek tidak dimiliki oleh Anda, pemilik ember. Kemudian, Anda dapat mendelegasikan izin kepada pengguna di akun Anda sendiri, atau ke orang lain Akun AWS. Misalkan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan S3 Object tidak diaktifkan. Itu berarti, bucket Anda dapat memiliki objek yang dimiliki Akun AWS lainnya.

Sekarang, seandainya sebagai pemilik bucket, Anda perlu memberikan izin lintas akun atas objek, terlepas dari siapa pemilik objek itu, kepada pengguna yang ada di akun lain. Sebagai contoh, pengguna tersebut bisa berupa aplikasi penagihan yang memerlukan akses terhadap metadata objek. Ada dua masalah inti:

- Pemilik bucket tidak memiliki izin pada objek yang dibuat oleh Akun AWS lain. Agar pemilik bucket dapat memberikan izin pada objek yang tidak dimilikinya, pemilik objek harus terlebih dahulu memberikan izin kepada pemilik bucket. Pemilik objek adalah Akun AWS yang menciptakan objek. Pemilik bucket dapat mendelegasikan izin tersebut.
- Akun pemilik bucket dapat mendelegasikan izin kepada pengguna di akunnya sendiri (lihat Contoh 3: Pemilik bucket yang memberikan izin kepada penggunanya ke objek yang bukan miliknya). Namun, akun pemilik bucket tidak dapat mendelegasikan izin ke akun lain Akun AWS karena delegasi lintas akun tidak didukung.

Dalam skenario ini, pemilik bucket dapat membuat peran AWS Identity and Access Management (IAM) dengan izin untuk mengakses objek Kemudian, pemilik bucket dapat memberikan Akun AWS izin lain untuk mengambil peran tersebut, untuk sementara memungkinkannya mengakses objek di bucket.

> **Note**
>
> Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda, serta menonaktifkan atau mengaktifkan ACL. Secara default, Kepemilikan Objek diatur ke pengaturan yang diberlakukan pemilik Bucket, dan semua ACL dinonaktifkan. Ketika ACL dinonaktifkan, pemilik bucket memiliki semua objek di bucket dan mengelola akses ke objek-objek tersebut secara eksklusif dengan menggunakan kebijakan manajemen akses. Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL. Kami menyarankan agar ACL dinonaktifkan, kecuali dalam keadaan yang tidak biasa ketika Anda perlu mengontrol akses untuk setiap objek secara individual. Dengan ACL dinonaktifkan, Anda dapat menggunakan kebijakan untuk mengontrol akses ke semua objek di bucket, terlepas dari siapa yang mengunggah objek ke bucket Anda. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Memahami izin lintas akun dan menggunakan peran IAM

Peran IAM memungkinkan beberapa skenario untuk mendelegasikan akses ke sumber daya Anda, dan akses lintas akun adalah salah satu skenario utamanya. Dalam contoh ini, pemilik bucket, Akun A, menggunakan peran IAM untuk sementara mendelegasikan akses objek lintas akun ke pengguna lain Akun AWS, Akun C. Setiap peran IAM yang Anda buat memiliki dua kebijakan berikut yang dilampirkan padanya:

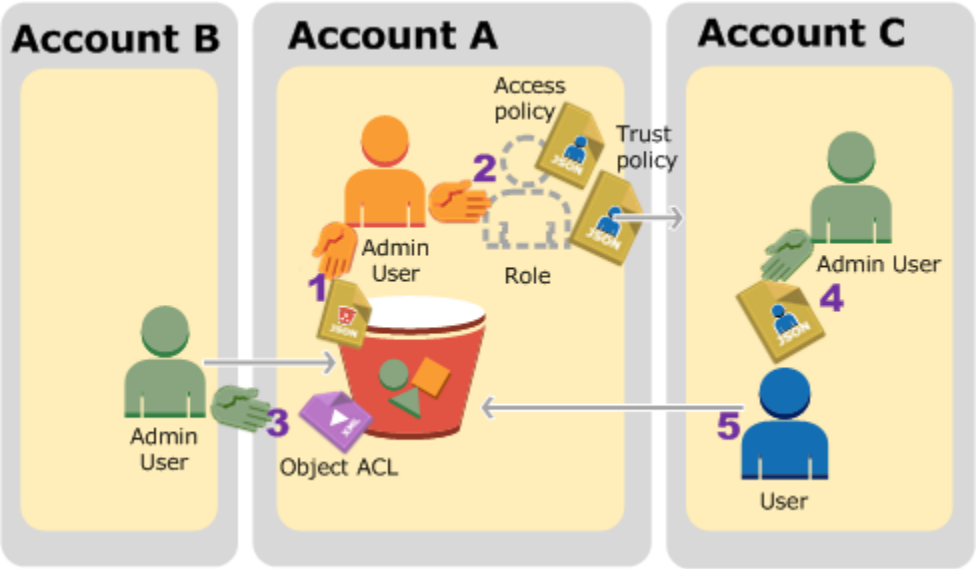
- Kebijakan kepercayaan mengidentifikasi orang lain Akun AWS yang dapat mengambil peran.
- Kebijakan akses yang menentukan izin—misalnya, `s3:GetObject`—yang diperkenankan saat seseorang mengambil peran tersebut. Untuk daftar izin yang dapat Anda tentukan dalam kebijakan, lihat Tindakan kebijakan untuk Amazon S3.

Yang Akun AWS diidentifikasi dalam kebijakan kepercayaan kemudian memberikan izin penggunanya untuk mengambil peran tersebut. Pengguna kemudian dapat melakukan hal-hal berikut untuk mengakses objek:

- Mengambil peran dan, sebagai tanggapan, mendapatkan kredensial keamanan sementara.
- Dengan menggunakan kredensial keamanan sementara, pengguna dapat mengakses objek yang ada dalam bucket.

Untuk informasi selengkapnya tentang peran IAM, lihat Peran IAM dalam Panduan Pengguna IAM.

Berikut ini adalah ringkasan langkah-langkah penelusuran:



1. Akun Pengguna administrator melampirkan kebijakan bucket yang memberikan izin bersyarat kepada Akun B untuk mengunggah objek.
2. Akun Administrator membuat peran IAM, membangun kepercayaan dengan Akun C, sehingga pengguna dalam akun tersebut dapat mengakses Akun A. Kebijakan akses yang terlampir pada peran tersebut membatasi apa yang dapat dilakukan pengguna yang ada di Akun C saat pengguna mengakses Akun A.
3. Administrator Akun B mengunggah objek ke bucket yang dimiliki oleh Akun A, memberikan izin kendali penuh kepada pemilik bucket.
4. Administrator Akun C membuat pengguna dan melampirkan kebijakan pengguna yang memungkinkan pengguna untuk mengambil peran tersebut.
5. Pengguna di Akun C terlebih dahulu harus mengambil peran tersebut, yang akan mengembalikan kredensial keamanan sementara kepada pengguna. Menggunakan kredensial sementara, pengguna kemudian dapat mengakses objek dalam bucket.

Untuk contoh ini, Anda memerlukan tiga akun. Tabel berikut menunjukkan cara kami merujuk akun-akun ini, dan pengguna administrator yang ada dalam akun-akun tersebut. Sesuai dengan pedoman IAM (lihatTentang menggunakan pengguna administrator untuk membuat sumber daya dan memberikan izin), kami tidak menggunakan Pengguna root akun AWS kredensyal dalam panduan ini. Sebagai gantinya, Anda membuat pengguna administrator di setiap akun dan menggunakan kredensial tersebut saat membuat sumber daya dan memberikan mereka izin.

Akun AWS ID	Akun disebut sebagai	Pengguna administrator di akun
1111-1111-1111	Akun A	AccountAadmin
2222-2222-2222	Akun B	AccountBadmin
3333-3333-3333	Akun C	AccountCadmin

Langkah 0: Bersiap-siap untuk panduan

> **Note**
>
> Anda mungkin ingin membuka editor teks, dan menuliskan beberapa informasi saat Anda melalui langkah-langkahnya. Secara kustom, Anda akan membutuhkan ID akun, ID pengguna kanonik, URL Masuk Pengguna IAM untuk setiap akun agar connect ke konsol tersebut, dan Amazon Resource Name (ARN) dari pengguna IAM, dan peran.

1. Pastikan Anda memiliki tiga Akun AWS dan setiap akun memiliki satu pengguna administrator seperti yang ditunjukkan pada tabel di bagian sebelumnya.

   a. Mendaftar untuk Akun AWS, sesuai kebutuhan. Kami menyebut akun ini sebagai Akun A, Akun B, dan Akun C.

   b. Menggunakan kredensial Akun A, masuklah ke Konsol IAM dan lakukan hal berikut ini untuk membuat pengguna administrator:

      - Buat pengguna **`AccountAadmin`** dan catat kredensi keamanannya. Untuk informasi lebih lanjut tentang penambahan pengguna, lihat Membuat Pengguna IAM dalam Akun AWS Anda dalam Panduan Pengguna IAM.
      - Berikan hak administrator AccountAadmindengan melampirkan kebijakan pengguna yang memberikan akses penuh. Untuk instruksi, lihat Mengelola kebijakan IAM dalam Panduan Pengguna IAM.
      - Di Dasbor Konsol IAM, perhatikan URL Masuk Pengguna IAM. Semua pengguna di akun tersebut harus menggunakan URL ini saat masuk ke AWS Management Console. Untuk informasi selengkapnya, lihat Masuk ke pengguna IAM AWS Management Console sebagai pengguna IAM di Panduan Pengguna IAM.

   c. Ulangi langkah-langkah sebelumnya untuk membuat pengguna administrator di Akun B dan Akun C.

2. Untuk Akun C, catat ID pengguna kanonik.

   Saat Anda membuat peran IAM di Akun A, kebijakan kepercayaan memberikan izin kepada Akun C untuk mengambil peran tersebut dengan menyebutkan ID akunnya. Anda dapat menemukan informasi akun seperti berikut ini:

a. Gunakan Akun AWS ID atau alias akun, nama pengguna IAM, dan kata sandi Anda untuk masuk ke konsol Amazon S3.

b. Pilih nama bucket Amazon S3 untuk melihat detail tentang bucket tersebut.

c. Pilih tab Izin dan kemudian pilih Daftar Kontrol Akses.

d. Pada bagian Akses Akun AWS Anda, pada kolom Akun adalah pengidentifikasi panjang, seperti `c1daexampleaaf850ea79cf0430f33d72579fd1611c97f7ded193374c0b163b6`. Ini adalah ID pengguna kanonis Anda.

3. Saat membuat kebijakan bucket, Anda akan memerlukan informasi berikut. Perhatikan nilai-nilai ini:

   - ID pengguna Kanonik Akun A–Ketika administrator Akun A memberikan izin unggah objek bersyarat kepada administrator Akun B, kondisinya menetapkan ID pengguna kanonik dari Akun A yang harus mendapatkan kontrol penuh atas objek.

     > **Note**
     >
     > ID pengguna kanonik adalah konsep Amazon S3 saja. ID ini adalah ID akun 64 karakter yang dikaburkan.

   - Pengguna ARN untuk Administrator Akun B — Anda dapat menemukan ARN pengguna di Konsol IAM. Anda harus memilih pengguna dan menemukan ARN pengguna di tab Ringkasan.

     Dalam kebijakan ember, Anda memberikan kepada `AccountBadmin` izin untuk mengunggah objek dan Anda menentukan pengguna dengan menggunakan ARN. Berikut ini adalah contoh nilai ARN:

     ```
     arn:aws:iam::AccountB-ID:user/AccountBadmin
     ```

4. Siapkan AWS Command Line Interface (CLI) atau. AWS Tools for Windows PowerShell Pastikan Anda menyimpan kredensi pengguna administrator sebagai berikut:

   - Jika menggunakan AWS CLI, buat profil, `AccountAadmin` dan`AccountBadmin`, dalam file konfigurasi.
   - Jika menggunakan AWS Tools for Windows PowerShell, pastikan bahwa Anda menyimpan kredensyal untuk sesi sebagai `AccountAadmin` dan. `AccountBadmin`

Untuk petunjuk, lihat Menyiapkan alat untuk penelusuran.

Langkah 1: Melakukan tugas Akun A

Dalam contoh ini, Akun A adalah pemilik bucket. Jadi pengguna AccountAadmin di Akun A akan melakukan hal berikut:

- Buat bucket.
- Lampirkan kebijakan bucket yang memberikan izin administrator Akun B untuk mengunggah objek.
- Buat peran IAM yang memberikan izin Akun C untuk mengambil peran sehingga dapat mengakses objek di bucket.

Langkah 1.1: Masuk ke AWS Management Console

Menggunakan URL Masuk pengguna IAM untuk Akun A, pertama-tama masuk ke pengguna AWS Management Console sebagai**`AccountAadmin`**. Pengguna ini akan membuat bucket dan melampirkan kebijakan pada bucket tersebut.

Langkah 1.2: Buat bucket dan melampirkan kebijakan bucket

Di konsol Amazon S3, lakukan berikut ini:

1. Buat bucket. Latihan ini mengasumsikan bahwa nama bucketnya adalah *`example-s3-bucket1`*.

   Untuk petunjuk, lihat Membuat bucket.
2. Lampirkan kebijakan bucket berikut. Kebijakan memberikan izin bersyarat kepada administrator Akun B untuk mengunggah objek.

   Perbarui kebijakan dengan memberikan nilai Anda sendiri untuk *`example-s3-bucket1AccountB-ID`*,, dan*`CanonicalUserId-of-AWSaccountA-BucketOwner`*.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "111",
            "Effect": "Allow",
```

```
            "Principal": {
                "AWS": "arn:aws:iam::AccountB-ID:user/AccountBadmin"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::example-s3-bucket1/*"
        },
        {
            "Sid": "112",
            "Effect": "Deny",
            "Principal": {
                "AWS": "arn:aws:iam::AccountB-ID:user/AccountBadmin"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::example-s3-bucket1/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-grant-full-control": "id=CanonicalUserId-of-
AWSaccountA-BucketOwner"
                }
            }
        }
    ]
}
```

Langkah 1.3: Buat peran IAM untuk memungkinkan akses lintas akun Akun C di Akun A

Di Konsol IAM, buat peran IAM (**`examplerole`**) yang memberikan izin Akun C untuk mengambil peran tersebut. Pastikan Anda masih masuk sebagai administrator Akun A karena peran harus dibuat di Akun A.

1. Sebelum membuat peran, siapkan kebijakan terkelola yang menentukan izin yang diperlukan oleh peran tersebut. Anda kemudian perlu melampirkan kebijakan ini ke peran tersebut pada langkah berikutnya.

   a. Di panel navigasi di sebelah kiri, pilih Kebijakan, lalu pilih Buat Kebijakan.

   b. Di sebelah Buat Kebijakan Anda Sendiri, pilih Pilih.

   c. Masukkan **`access-accountA-bucket`** dalam kolom Nama Kebijakan.

   d. Salin kebijakan akses berikut ini dan tempelkan ke kolom Dokumen Kebijakan. Kebijakan akses memberikan `s3:GetObject` izin peran sehingga, ketika pengguna Akun C mengambil peran, kebijakan tersebut hanya dapat melakukan operasi. `s3:GetObject`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::example-s3-bucket1/*"
    }
  ]
}
```

e. Pilih Buat Kebijakan.

   Kebijakan baru muncul di daftar kebijakan terkelola.

2. Di panel navigasi di sebelah kiri, pilih Peran dan kemudian pilih Buat Peran Baru.
3. Di bawah Pilih Jenis Peran, pilih Peran untuk Akses Lintas Akun, lalu pilih tombol Pilih di samping Menyediakan akses di antara Akun AWS Anda sendiri.
4. Masukkan ID akun dari Akun C.

   Untuk panduan ini, Anda tidak perlu meminta pengguna untuk memiliki otentikasi multi-faktor (MFA) untuk mengambil peran, jadi biarkan opsi itu tidak dipilih.
5. Pilih Langkah Berikutnya untuk mengatur izin yang akan dikaitkan dengan peran.
6. Pilih kotak centang di samping kebijakan Access-Accounta-bucket yang Anda buat, lalu pilih Langkah Berikutnya.

   Halaman Tinjauan muncul sehingga Anda dapat mengonfirmasi pengaturan untuk peran tersebut sebelum dibuat. Satu item yang sangat penting untuk dicatat di halaman ini adalah tautan yang dapat Anda kirim kepada pengguna Anda yang perlu menggunakan peran ini. Pengguna yang menggunakan tautan langsung menuju halaman Beralih Peran dengan kolom ID Akun dan Nama Peran yang sudah diisi. Anda juga dapat melihat tautan ini nanti di halaman Ringkasan Peran untuk peran lintas akun apa pun.
7. Masukkan `examplerole` nama peran, lalu pilih Langkah Berikutnya.
8. Setelah meninjau peran, pilih Buat Peran.

   Peran `examplerole` ditampilkan dalam daftar peran.
9. Pilih nama peran`examplerole`.

10. Pilih tab Hubungan Kepercayaan.
11. Pilih Tampilkan dokumen kebijakan dan verifikasi kebijakan kepercayaan yang ditampilkan cocok dengan kebijakan berikut.

    Kebijakan kepercayaan berikut ini membangun kepercayaan terhadap Akun C, dengan memungkinkannya untuk melakukan tindakan `sts:AssumeRole`. Untuk informasi selengkapnya, lihat AssumeRole di dalam Referensi API AWS Security Token Service .

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::AccountC-ID:root"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

12. Perhatikan Nama Sumber Daya Amazon (ARN) dari `examplerole` peran yang Anda buat.

    Kemudian pada langkah-langkah berikut ini, Anda melampirkan kebijakan pengguna untuk memungkinkan pengguna IAM mengambil peran ini, dan kemudian Anda mengidentifikasi peran berdasarkan nilai ARN.

Langkah 2: Melakukan tugas Akun B

Contoh bucket yang dimiliki oleh Akun A memerlukan objek yang dimiliki oleh akun lain. Pada langkah ini, administrator Akun B mengunggah objek dengan menggunakan alat baris perintah.

- Menggunakan `put-object` AWS CLI perintah, unggah objek ke*`example-s3-bucket1`*.

```
aws s3api put-object --bucket example-s3-bucket1 --key HappyFace.jpg --
body HappyFace.jpg --grant-full-control id="canonicalUserId-ofTheBucketOwner" --
profile AccountBadmin
```

  Perhatikan hal berikut:

- `--Profile`Parameter menentukan `AccountBadmin` profil, sehingga objek dimiliki oleh Akun B.
- Parameter `grant-full-control` memberikan izin kendali penuh kepada pemilik bucket pada objek sebagaimana yang diwajibkan oleh kebijakan bucket.
- Parameter `--body` mengidentifikasi file sumber yang akan diunggah. Misalnya, jika file tersebut ada di drive C: Windows komputer, Anda tentukan`c:\HappyFace.jpg`.

Langkah 3: Lakukan tugas Akun C

Pada langkah-langkah sebelumnya, Akun A telah menciptakan peran`examplerole`, membangun kepercayaan dengan Akun C. Peran ini memungkinkan pengguna di Akun C untuk mengakses Akun A. Pada langkah ini, administrator Akun C membuat pengguna (Dave) dan mendelegasikan kepadanya `sts:AssumeRole` izin yang diterima dari Akun A. Pendekatan ini memungkinkan Dave untuk mengasumsikan `examplerole` dan untuk sementara mendapatkan akses ke Akun A. Kebijakan akses yang Akun A melekat pada peran membatasi apa yang dapat dilakukan Dave ketika dia mengakses Akun A—khususnya, masukkan objek. *`example-s3-bucket1`*

Langkah 3.1: Buat pengguna di Akun C dan delegasikan izin untuk berasumsi examplerole

1. Menggunakan URL login pengguna IAM untuk Akun C, pertama-tama masuk ke pengguna AWS Management Console sebagai**`AccountCadmin`**.

2. Di Konsol IAM, buat pengguna, Dave.

   Untuk step-by-step petunjuk, lihat Membuat pengguna IAM (AWS Management Console) di Panduan Pengguna IAM.
3. Perhatikan kredensyal Dave. Dave akan memerlukan kredensial ini untuk mengambil peran `examplerole`.
4. Buat kebijakan inline untuk pengguna Dave IAM untuk mendelegasikan `sts:AssumeRole` izin kepada Dave pada peran di Akun A. `examplerole`

   a. Pada panel navigasi di sebelah kiri, pilih Pengguna.

   b. Pilih nama pengguna Dave.

   c. Pada halaman rincian pengguna, pilih tab Izin dan kemudian perluas bagian Kebijakan Selaras.

   d. Pilih klik di sini (atau Buat Kebijakan Pengguna).

e. Pilih Kebijakan Kustom, lalu pilih Pilihan.

f. Masukkan nama untuk kebijakan pada kolom Nama Kebijakan.

g. Salin kebijakan berikut ke kolom Dokumen Kebijakan.

Anda harus memperbarui kebijakan dengan menyediakan*AccountA-ID*.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["sts:AssumeRole"],
      "Resource": "arn:aws:iam::AccountA-ID:role/examplerole"
    }
  ]
}
```

h. Pilih Terapkan Kebijakan.

5. Simpan kredensi Dave ke file konfigurasi AWS CLI dengan menambahkan profil lain,. `AccountCDave`

```
[profile AccountCDave]
aws_access_key_id = UserDaveAccessKeyID
aws_secret_access_key = UserDaveSecretAccessKey
region = us-west-2
```

Langkah 3.2: Asumsikan peran (examplerole) dan akses objek

Kini Dave dapat mengakses objek yang ada dalam bucket yang dimiliki oleh Akun A seperti berikut ini:

- Dave terlebih dahulu harus mengambil `examplerole` menggunakan kredensialnya sendiri. Hal ini akan mengembalikan kredensial sementara.
- Dengan menggunakan kredensial sementara, Dave kemudian akan dapat mengakses objek yang ada dalam bucket Akun A.

1. Pada prompt perintah, jalankan AWS CLI `assume-role` perintah berikut menggunakan `AccountCDave` profil.

Anda harus memperbarui nilai ARN dalam perintah dengan memberikan di *AccountA-ID* mana `examplerole` didefinisikan.

```
aws sts assume-role --role-arn arn:aws:iam::AccountA-ID:role/examplerole --profile
 AccountCDave --role-session-name test
```

Sebagai tanggapan, AWS Security Token Service (AWS STS) mengembalikan kredensyal keamanan sementara (ID kunci akses, kunci akses rahasia, dan token sesi).

2. Simpan kredensil keamanan sementara di file AWS CLI konfigurasi di bawah profil. `TempCred`

```
[profile TempCred]
aws_access_key_id = temp-access-key-ID
aws_secret_access_key = temp-secret-access-key
aws_session_token = session-token
region = us-west-2
```

3. Pada prompt perintah, jalankan AWS CLI perintah berikut untuk mengakses objek menggunakan kredensyal sementara. Misalnya, perintah menentukan head-object API untuk mengambil metadata objek untuk `HappyFace.jpg` objek.

```
aws s3api get-object --bucket example-s3-bucket1 --key HappyFace.jpg SaveFileAs.jpg
 --profile TempCred
```

Karena kebijakan akses terlampir pada `examplerole` memungkinkan tindakan, Amazon S3 memproses permintaan. Anda dapat mencoba tindakan lain apa pun pada objek lain yang ada dalam bucket.

Jika Anda mencoba tindakan lain—misalnya, `get-object-acl` —Anda akan mendapatkan izin ditolak karena peran tersebut tidak diizinkan tindakan itu.

```
aws s3api get-object-acl --bucket example-s3-bucket1 --key HappyFace.jpg --profile
 TempCred
```

Kita menggunakan pengguna Dave untuk mengambil peran dan mengakses objek dengan menggunakan kredensial sementara. Ini juga bisa menjadi aplikasi di Akun C yang mengakses objek di *example-s3-bucket1*. Aplikasi dapat memperoleh kredensial keamanan sementara, dan Akun C dapat mendelegasikan izin aplikasi untuk mengambil `examplerole`.

Langkah 4: Membersihkan

1. Setelah selesai melakukan pengujian, Anda dapat melakukan hal berikut untuk membersihkan:
   - Masuk ke AWS Management Console menggunakan kredensial Akun A, dan lakukan hal-hal berikut ini:
     - Pada konsol Amazon S3, hapus kebijakan bucket yang terlampir pada *`example-s3-bucket1`*. Dalam bucket Properti, hapus kebijakan yang ada dalam bagian Izin.
     - Jika bucket dibuat untuk latihan ini, pada konsol Amazon S3, hapus objek dan kemudian hapus bucket.
     - Di Konsol IAM, hapus yang `examplerole` Anda buat di Akun A. Untuk step-by-step petunjuk, lihat Menghapus pengguna IAM di Panduan Pengguna IAM.
     - Di Konsol IAM, hapus AccountAadminpengguna.
2. Masuk ke Konsol IAM menggunakan kredensyal Akun B. Hapus pengguna AccountBadmin.
3. Masuk ke Konsol IAM dengan menggunakan kredensyal Akun C. Hapus AccountCadmindan pengguna Dave.

Sumber daya terkait

Untuk informasi selengkapnya yang terkait dengan panduan ini, lihat sumber daya berikut di Panduan Pengguna IAM:

- Membuat peran untuk mendelegasikan izin ke pengguna IAM
- Tutorial: Mendelegasikan Akses di Seluruh Akun AWS Menggunakan Peran IAM
- Mengelola kebijakan IAM

Bagaimana Amazon S3 Mengotorisasi Permintaan

Ketika Amazon S3 menerima permintaan—misalnya, operasi bucket atau operasi objek—pertama-tama akan memverifikasi apakah pemohon memiliki izin yang diperlukan. Amazon S3 mengevaluasi semua kebijakan akses yang relevan, kebijakan pengguna, dan kebijakan berbasis sumber daya (kebijakan bucket, daftar kontrol akses bucket (ACL), dan objek ACL) dalam memutuskan apakah akan mengotorisasi permintaan.

> **Note**
>
> Jika pemeriksaan izin Amazon S3 gagal menemukan izin yang valid, kesalahan akses ditolak (403 Terlarang) akan dikembalikan. Untuk informasi selengkapnya, lihat Memecahkan masalah kesalahan Akses Ditolak (403 Terlarang) di Amazon S3.

Untuk menentukan apakah pemohon memiliki izin untuk melakukan operasi tertentu, Amazon S3 melakukan hal berikut, secara berurutan, saat menerima permintaan:

1. Mengonversi semua kebijakan akses yang relevan (kebijakan pengguna, kebijakan bucket, dan ACL) pada waktu berjalan menjadi serangkaian kebijakan untuk evaluasi.
2. Mengevaluasi serangkaian kebijakan yang dihasilkan dalam langkah-langkah berikut. Dalam setiap langkah, Amazon S3 akan mengevaluasi sebagian dari kebijakan dalam konteks tertentu, berdasarkan otoritas konteks.
   a. Konteks pengguna–Dalam konteks pengguna, akun induk yang merupakan pemilik pengguna adalah otoritas konteks.

      Amazon S3 mengevaluasi sebagian kebijakan yang dimiliki oleh akun induk. Sebagian kebijakan ini mencakup kebijakan pengguna yang dilampirkan oleh akun induk kepada pengguna. Jika induk juga memiliki sumber daya dalam permintaan (bucket atau objek), Amazon S3 juga mengevaluasi kebijakan sumber daya terkait (kebijakan bucket, bucket ACL, dan objek ACL) secara bersamaan.

      Pengguna harus memiliki izin dari akun induk untuk melakukan operasi.

      Langkah ini hanya berlaku jika permintaan diajukan oleh pengguna dalam Akun AWS. Jika permintaan dibuat dengan menggunakan kredensyal pengguna root dari sebuah Akun AWS, Amazon S3 melewatkan langkah ini.

b. Konteks bucket — Dalam konteks bucket, Amazon S3 mengevaluasi kebijakan yang dimiliki oleh pemilik Akun AWS bucket.

   Apabila permintaan digunakan untuk operasi bucket, maka pemohon harus memiliki izin dari pemilik bucket. Jika permintaan untuk sebuah objek, maka Amazon S3 akan mengevaluasi semua kebijakan yang dimiliki oleh pemilik bucket untuk memeriksa apakah pemilik bucket belum secara jelas menolak akses ke objek tersebut. Jika ada penolakan jelas yang ditetapkan, Amazon S3 tidak mengotorisasi permintaan.

c. Konteks objek–Jika permintaan untuk objek, maka Amazon S3 akan mengevaluasi sebagian kebijakan yang dimiliki oleh pemilik objek.

Berikut adalah beberapa contoh skenario yang menggambarkan bagaimana Amazon S3 mengotorisasi permintaan.

Example — Pemohon adalah prinsipal IAM

Jika pemohon adalah prinsipal IAM, Amazon S3 harus menentukan apakah Akun AWS induk tempat prinsipal tersebut telah memberikan izin pokok yang diperlukan untuk melakukan operasi. Selain itu, jika permintaan tersebut untuk operasi bucket, seperti permintaan untuk mencantumkan konten bucket, maka Amazon S3 harus memverifikasi bahwa pemilik bucket telah memberikan izin bagi pemohon untuk melakukan operasi tersebut. Untuk melakukan operasi tertentu pada sumber daya, prinsipal IAM memerlukan izin dari induk yang Akun AWS menjadi miliknya dan Akun AWS yang memiliki sumber daya tersebut.

Example — Pemohon adalah prinsipal IAM — Jika permintaan adalah untuk operasi pada objek yang tidak dimiliki pemilik bucket

Jika permintaan adalah untuk operasi pada objek yang tidak dimiliki pemilik bucket, selain memastikan pemohon memiliki izin dari pemilik objek, Amazon S3 juga harus memeriksa kebijakan bucket untuk memastikan pemilik bucket belum menetapkan penolakan eksplisit pada objek. Pemilik bucket (yang membayar tagihan) dapat dengan tegas menolak akses ke objek di dalam bucket, terlepas dari siapa yang memilikinya. Pemilik bucket juga dapat menghapus objek apa pun yang ada dalam bucket.

Secara default, ketika orang lain Akun AWS mengunggah objek ke bucket S3 Anda, akun tersebut (penulis objek) memiliki objek tersebut, memiliki akses ke objek tersebut, dan dapat memberikan akses kepada pengguna lain melalui daftar kontrol akses (ACL). Anda dapat menggunakan

Kepemilikan Objek untuk mengubah perilaku default ini sehingga ACL dinonaktifkan dan Anda, sebagai pemilik bucket, secara otomatis memiliki setiap objek di bucket Anda. Akibatnya, kontrol akses untuk data Anda didasarkan pada kebijakan, seperti kebijakan pengguna IAM, kebijakan bucket S3, kebijakan titik akhir virtual private cloud (VPC), dan kebijakan kontrol AWS Organizations layanan (SCP). Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Untuk informasi lebih lanjut tentang bagaimana Amazon S3 mengevaluasi kebijakan akses untuk mengotorisasi atau tolak permintaan untuk operasi bucket dan operasi objek, lihat topik berikut:

Topik

- Bagaimana Amazon S3 Mengotorisasi Permintaan Operasi bucket
- Bagaimana Amazon S3 Memberikan Otorisasi Permintaan Operasi Objek

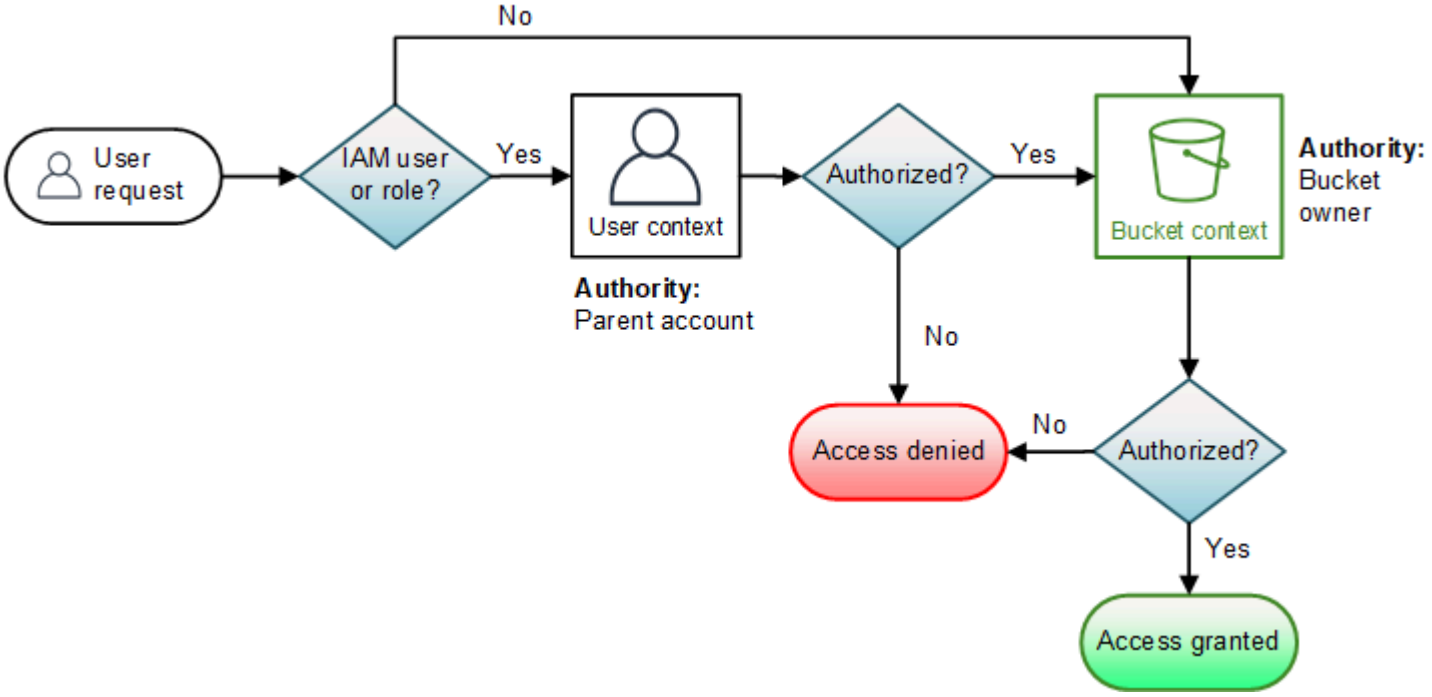
Bagaimana Amazon S3 Mengotorisasi Permintaan Operasi bucket

Ketika Amazon S3 menerima permintaan untuk operasi bucket, Amazon S3 mengubah semua izin yang relevan menjadi serangkaian kebijakan untuk mengevaluasi pada runtime. Izin yang relevan termasuk izin berbasis sumber daya (misalnya, kebijakan bucket dan daftar kontrol akses bucket) dan kebijakan pengguna IAM jika permintaan dari pengguna utama IAM. Amazon S3 kemudian mengevaluasi serangkaian kebijakan yang dihasilkan dalam serangkaian langkah sesuai dengan konteks tertentu—konteks pengguna atau konteks bucket:

1. Konteks pengguna - Jika pemohon adalah prinsipal IAM, kepala sekolah harus memiliki izin dari orang tua yang Akun AWS menjadi miliknya. Dalam langkah ini, Amazon S3 mengevaluasi sebagian kebijakan yang dimiliki oleh akun induk (juga disebut sebagai otoritas konteks). Bagian dari kebijakan ini mencakup kebijakan pengguna yang dilampirkan oleh akun induk pada pengguna utama. Jika induk juga memiliki sumber daya dalam permintaan (dalam hal ini, bucket), maka Amazon S3 juga akan mengevaluasi kebijakan sumber daya yang sesuai (kebijakan bucket dan ACL bucket) pada waktu yang bersamaan. Setiap kali permintaan untuk operasi bucket dibuat, log akses server akan mencatat ID kanonik dari pemohon. Untuk informasi selengkapnya, lihat Pencatatan permintaan dengan pencatatan akses server.
2. Konteks bucket Pemohon harus memiliki izin dari pemilik bucket untuk melakukan operasi bucket tertentu. Pada langkah ini, Amazon S3 mengevaluasi subset kebijakan yang dimiliki oleh pemilik bucket Akun AWS .

Pemilik bucket dapat memberikan izin dengan menggunakan kebijakan bucket atau ACL bucket. Jika Akun AWS yang memiliki bucket juga merupakan akun induk dari prinsipal IAM, maka bucket tersebut dapat mengonfigurasi izin bucket dalam kebijakan pengguna.

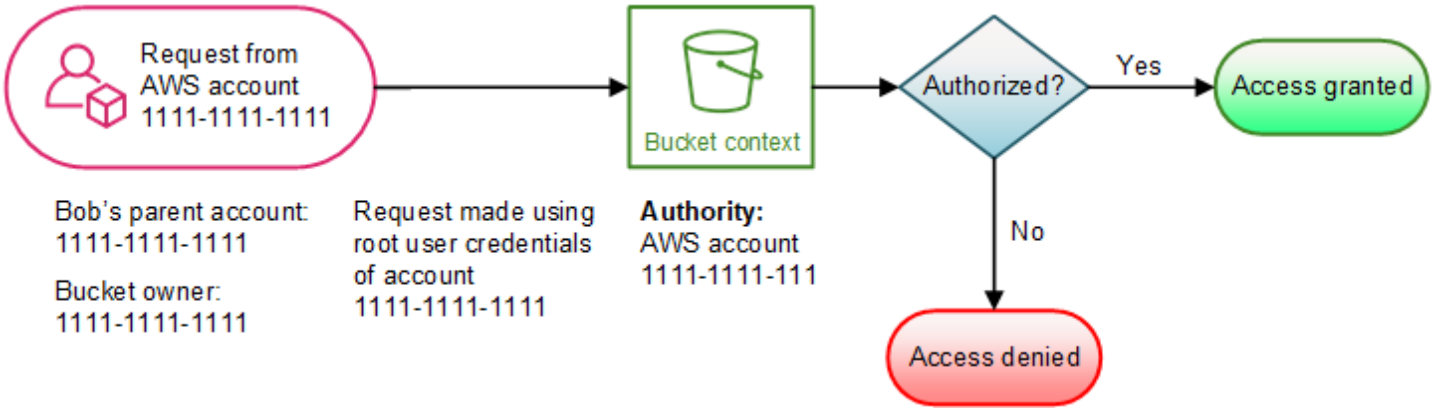
Berikut ini adalah ilustrasi grafis evaluasi berbasis konteks untuk operasi bucket.



Contoh-contoh berikut ini menggambarkan logika evaluasi.

Contoh 1: Operasi bucket Diminta oleh pemilik bucket

Dalam contoh ini, pemilik bucket mengirimkan permintaan operasi bucket dengan menggunakan kredensial root Akun AWS.

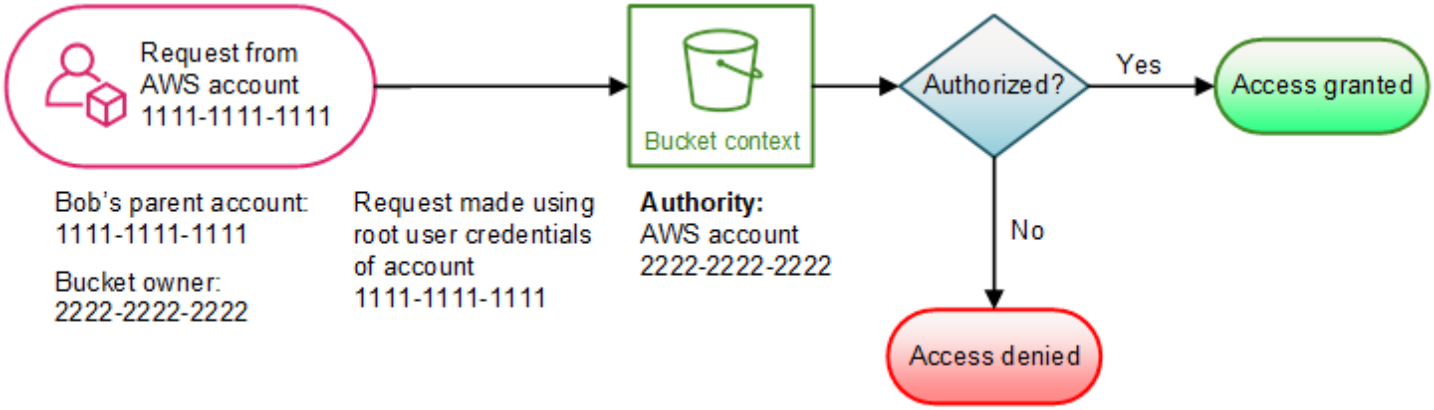


Amazon S3 melakukan evaluasi konteks sebagai berikut:

1. Karena permintaan dibuat dengan menggunakan kredensial pengguna root Akun AWS, maka konteks pengguna tidak dievaluasi.
2. Dalam konteks bucket, Amazon S3 akan meninjau kebijakan bucket untuk menentukan apakah pemohon memiliki izin untuk melakukan operasi atau tidak. Amazon S3 mengotorisasi permintaan tersebut.

Contoh 2: Operasi bucket yang diminta oleh pemilik bucket Akun AWS yang bukan pemilik bucket

Dalam contoh ini, permintaan dibuat dengan menggunakan kredensial pengguna root Akun AWS 1111-1111-1111 untuk operasi bucket yang dimiliki oleh 2222-2222-2222 Akun AWS . Tidak ada pengguna IAM yang dilibatkan dalam permintaan ini.

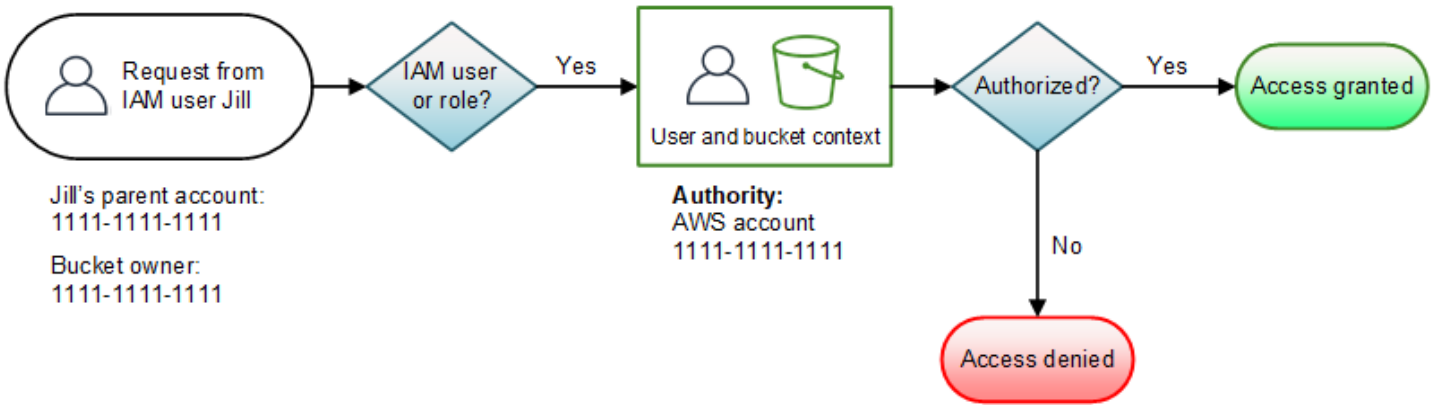


Dalam contoh ini, Amazon S3 mengevaluasi konteksnya sebagai berikut:

1. Karena permintaan dibuat dengan menggunakan kredensyal pengguna root dari sebuah Akun AWS, konteks pengguna tidak dievaluasi.
2. Dalam konteks bucket, Amazon S3 memeriksa kebijakan bucket. Jika pemilik bucket (Akun AWS 2222-2222-2222) belum mengizinkan Akun AWS 1111-1111-1111 untuk melakukan operasi yang diminta, Amazon S3 menolak permintaan tersebut. Jika tidak, Amazon S3 akan memberikan permintaan dan melakukan operasi.

Contoh 3: Operasi bucket yang diminta oleh kepala sekolah IAM yang Akun AWS induknya juga pemilik bucket

Sebagai contoh, permintaan dikirim oleh Jill, pengguna IAM di Akun AWS 1111-1111-1111, yang juga merupakan pemilik bucket.



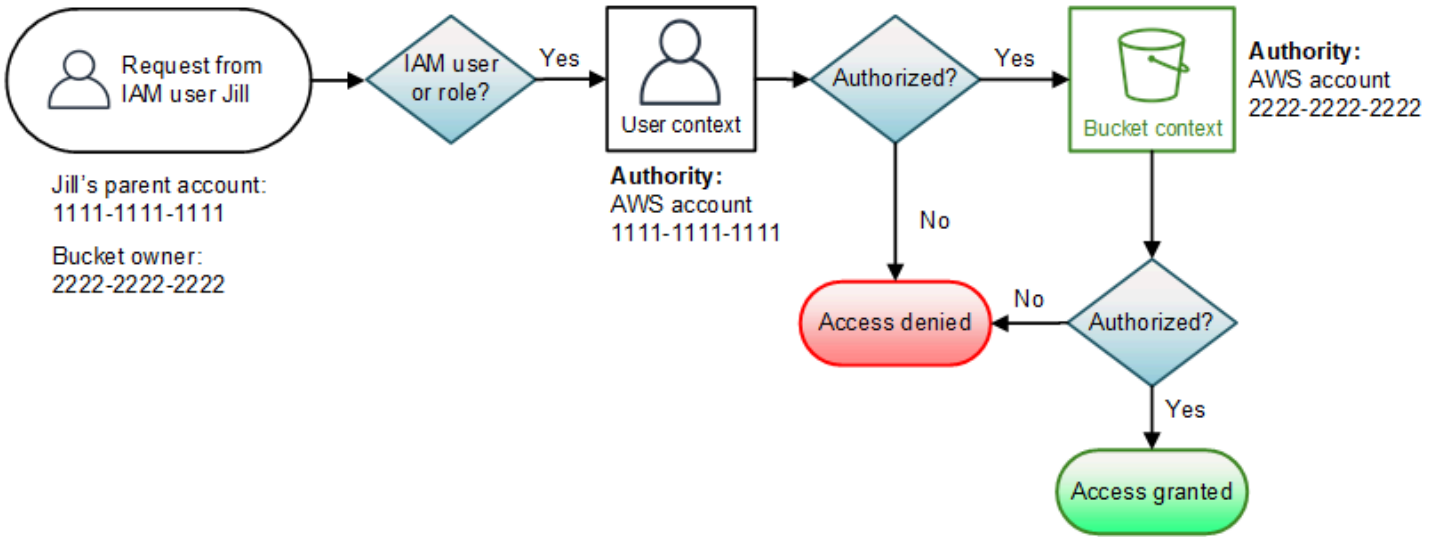
Amazon S3 akan melakukan evaluasi konteks sebagai berikut:

1. Karena permintaan berasal dari prinsipal IAM, dalam konteks pengguna, Amazon S3 mengevaluasi semua kebijakan milik Akun AWS induk untuk menentukan apakah Jill memiliki izin untuk melakukan operasi.

   Dalam contoh ini, induk Akun AWS 1111-1111-1111, yang menjadi milik prinsipal, juga merupakan pemilik ember. Akibatnya, selain kebijakan pengguna, Amazon S3 juga mengevaluasi kebijakan bucket dan bucket ACL dalam konteks yang sama karena mereka termasuk dalam akun yang sama.
2. Karena Amazon S3 mengevaluasi kebijakan bucket dan ACL bucket sebagai bagian dari konteks pengguna, maka itu tidak akan mengevaluasi konteks bucket.

Contoh 4: Operasi bucket yang diminta oleh kepala sekolah IAM yang induknya Akun AWS bukan pemilik bucket

Dalam contoh ini, permintaan dikirim oleh Jill, pengguna IAM yang induknya Akun AWS adalah 1111-1111-1111, tetapi ember tersebut dimiliki oleh yang lain, 2222-2222-2222. Akun AWS



Jill akan membutuhkan izin dari induk Akun AWS dan pemilik ember. Amazon S3 mengevaluasi konteks seperti berikut ini:

1. Karena permintaan tersebut berasal dari pengguna utama IAM, Amazon S3 melakukan evaluasi konteks pengguna dengan meninjau kebijakan yang ditulis oleh akun tersebut untuk memverifikasi bahwa Jill memiliki izin yang diperlukan. Jika Jill memiliki izin, Amazon S3 melanjutkan untuk mengevaluasi konteks bucket. Jika Jill tidak memiliki izin, ia menolak permintaan tersebut.
2. Dalam konteks bucket, Amazon S3 memverifikasi bahwa pemilik bucket 2222-2222-2222 telah memberikan izin kepada Jill (atau orang tuanya) untuk melakukan operasi yang diminta. Akun AWS Jika dia memiliki izin itu, Amazon S3 memberikan permintaan dan melakukan operasi. Jika tidak, maka Amazon S3 akan menolak permintaan tersebut.

Bagaimana Amazon S3 Memberikan Otorisasi Permintaan Operasi Objek

Saat Amazon S3 menerima permintaan untuk operasi objek, itu akan mengubah semua izin yang relevan— izin berbasis sumber daya (daftar kontrol akses (ACL) objek, kebijakan bucket, ACL bucket) dan kebijakan pengguna IAM—menjadi serangkaian kebijakan yang akan dievaluasi pada waktu berjalan. Kemudian, itu akan mengevaluasi serangkaian kebijakan yang dihasilkan dengan serangkaian langkah. Dalam setiap langkah, ia mengevaluasi subset kebijakan dalam tiga konteks tertentu—konteks pengguna, konteks bucket, dan konteks objek:

1. Konteks pengguna - Jika pemohon adalah prinsipal IAM, kepala sekolah harus memiliki izin dari orang tua yang Akun AWS menjadi miliknya. Dalam langkah ini, Amazon S3 mengevaluasi sebagian kebijakan yang dimiliki oleh akun induk (juga disebut sebagai otoritas konteks). Sebagian dari kebijakan ini mencakup kebijakan pengguna yang dilampirkan oleh akun induk pada

pengguna utama. Jika induk juga memiliki sumber daya dalam permintaan (bucket atau objek), Amazon S3 mengevaluasi kebijakan sumber daya terkait (kebijakan bucket, bucket ACL, dan objek ACL) secara bersamaan.

> **Note**
>
> Jika Akun AWS induk memiliki sumber daya (bucket atau objek), ia dapat memberikan izin sumber daya ke prinsipal IAM-nya dengan menggunakan kebijakan pengguna atau kebijakan sumber daya.

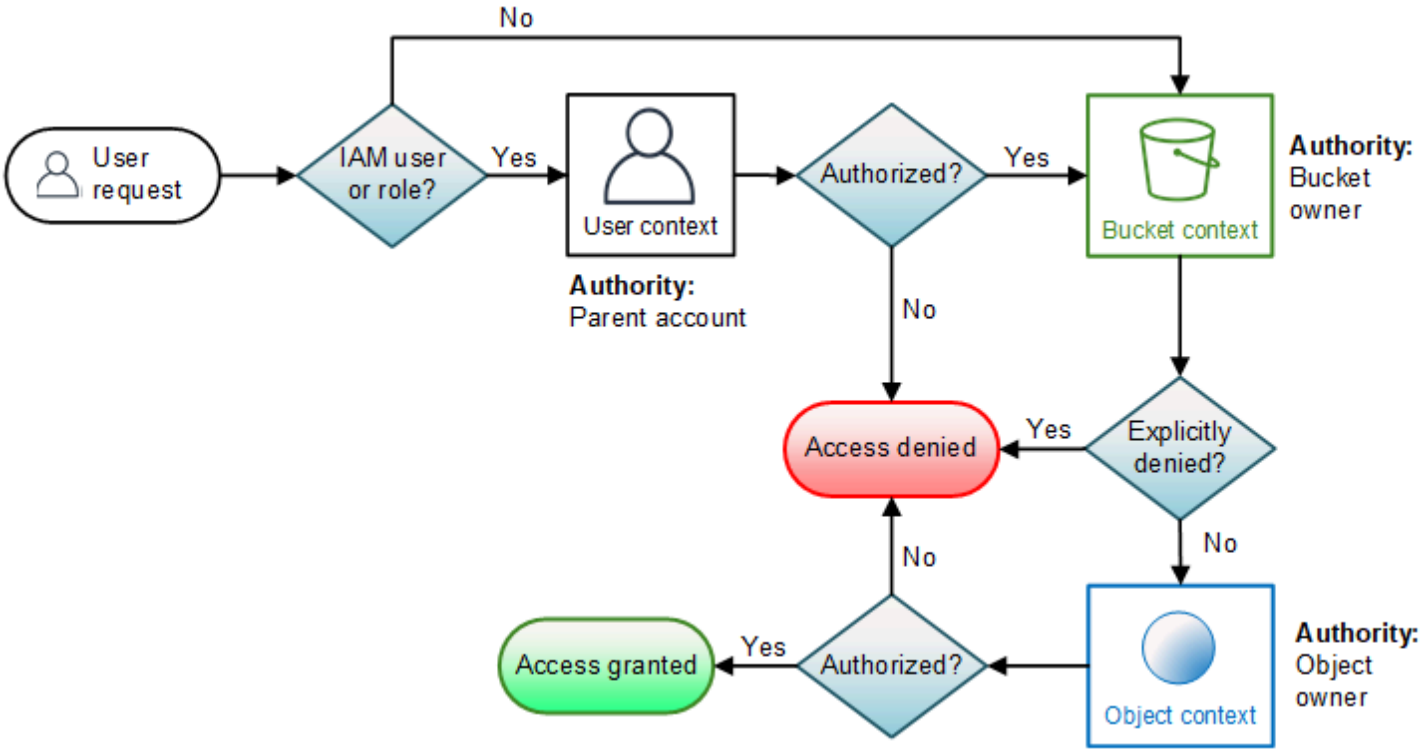
2. Konteks bucket–Dalam konteks ini, Amazon S3 mengevaluasi kebijakan yang dimiliki oleh Akun AWS yang memiliki bucket tersebut.

   Jika pemilik Akun AWS objek dalam permintaan tidak sama dengan pemilik bucket, Amazon S3 akan memeriksa kebijakan jika pemilik bucket secara eksplisit menolak akses ke objek tersebut. Jika ada penolakan tegas yang ditetapkan pada objek, Amazon S3 tidak mengizinkan permintaan tersebut.

3. Konteks objek–Pemohon harus memiliki izin dari pemilik objek untuk melakukan operasi objek tertentu. Dalam langkah ini, Amazon S3 mengevaluasi ACL objek.

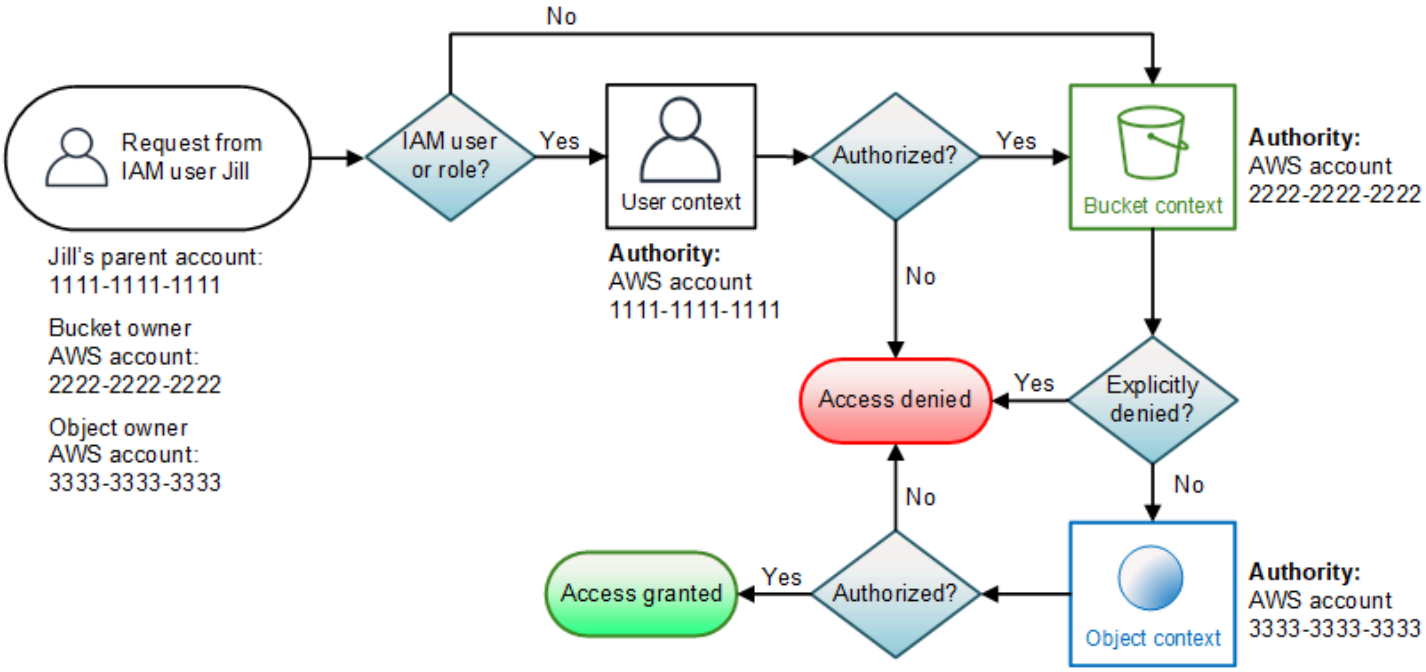
> **Note**
>
> Jika pemilik bucket dan objek sama, maka akses ke objek dapat diberikan dalam kebijakan bucket, yang dievaluasi dalam konteks bucket. Jika pemiliknya berbeda, maka pemilik objek harus menggunakan ACL objek untuk memberikan izin. Jika Akun AWS yang memiliki objek juga merupakan akun induk tempat prinsipal IAM berada, ia dapat mengonfigurasi izin objek dalam kebijakan pengguna, yang dievaluasi pada konteks pengguna. Untuk informasi lebih lanjut tentang penggunaan berbagai alternatif kebijakan akses ini, lihat Panduan yang menggunakan kebijakan untuk mengelola akses ke sumber daya Amazon S3.
> Jika Anda sebagai pemilik bucket ingin memiliki semua objek di bucket dan menggunakan kebijakan atau kebijakan bucket berdasarkan IAMUntuk mengelola akses ke objek ini, Anda dapat menerapkan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek. Dengan pengaturan ini, Anda sebagai pemilik bucket secara otomatis memiliki dan memiliki kontrol penuh atas setiap objek di bucket Anda. Bucket dan ACL objek tidak dapat diedit dan tidak lagi dipertimbangkan untuk diakses. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Berikut ini adalah ilustrasi dari evaluasi berbasis konteks untuk operasi objek.



Contoh permintaan operasi objek

Dalam contoh ini, pengguna IAM Jill, yang induknya Akun AWS adalah 1111-1111-1111, mengirimkan permintaan operasi objek (misalnya,`GetObject`) untuk objek yang dimiliki oleh Akun AWS 3333-3333-3333 dalam ember yang dimiliki oleh 2222-2222-2222. Akun AWS



Jill akan membutuhkan izin dari orang tua Akun AWS, pemilik ember, dan pemilik objek. Amazon S3 mengevaluasi konteksnya seperti berikut:

1. Karena permintaan tersebut berasal dari prinsipal IAM, Amazon S3 mengevaluasi konteks pengguna untuk memverifikasi bahwa Akun AWS induk 1111-1111-1111 telah memberikan izin kepada Jill untuk melakukan operasi yang diminta. Jika dia memiliki izin tersebut, maka Amazon S3 akan mengevaluasi konteks bucket. Jika tidak, maka Amazon S3 akan menolak permintaan tersebut.
2. Dalam konteks bucket, pemilik bucket, Akun AWS 2222-2222-2222, adalah otoritas konteks. Amazon S3 mengevaluasi kebijakan bucket untuk menentukan apakah pemilik bucket telah secara jelas menolak akses Jill ke objek.
3. Dalam konteks objek, otorisasi konteks adalah Akun AWS 3333-3333-3333, pemilik objek. Amazon S3 mengevaluasi ACL objek untuk menentukan apakah Jill memiliki izin untuk mengakses objek tersebut. Jika ya, maka Amazon S3 akan mengesahkan permintaan tersebut.

AWS kebijakan terkelola untuk Amazon S3

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS Kebijakan terkelola dirancang untuk memberikan izin bagi banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan semua pelanggan. AWS Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pemutakhiran akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan AWS terkelola saat baru Layanan AWS diluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat AWS kebijakan yang dikelola dalam Panduan Pengguna IAM.

AWS kebijakan terkelola: AmazonS3FullAccess

Anda dapat melampirkan kebijakan `AmazonS3FullAccess` ke identitas IAM Anda. Kebijakan ini akan memberikan izin yang mengizinkan akses penuh ke Amazon S3.

Untuk menampilkan izin untuk kebijakan ini, lihat AmazonS3FullAccess di AWS Management Console.

AWS kebijakan terkelola: AmazonS3ReadOnlyAccess

Anda dapat melampirkan kebijakan `AmazonS3ReadOnlyAccess` ke identitas IAM Anda. Kebijakan ini memberikan izin yang memungkinkan akses hanya-baca ke Amazon S3.

Untuk menampilkan izin untuk kebijakan ini, lihat AmazonS3ReadOnlyAccess di AWS Management Console.

Kebijakan terkelola AWS : AmazonS3ObjectLambdaExecutionRolePolicy

Menyediakan AWS Lambda fungsi izin yang diperlukan untuk mengirim data ke Lambda Objek S3 saat permintaan dibuat ke titik akses Lambda Objek S3. Juga memberikan izin Lambda untuk menulis ke log Amazon. CloudWatch

Untuk menampilkan izin untuk kebijakan ini, lihat AmazonS3ObjectLambdaExecutionRolePolicy di AWS Management Console.

Pembaruan Amazon S3 ke AWS kebijakan terkelola

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk Amazon S3 sejak layanan ini mulai melacak perubahan ini.

Perubahan	Deskripsi	Tanggal
Amazon S3 menambahkan izin Jelaskan ke `AmazonS3ReadOnlyAccess`	Amazon S3 menambahkan izin `s3:Describe*` ke `AmazonS3ReadOnlyAccess` .	11 Agustus 2023
Amazon S3 menambahkan izin S3 Lambda Objek ke `AmazonS3FullAccess` dan `AmazonS3ReadOnlyAccess`	Amazon S3 memperbarui kebijakan `AmazonS3FullAccess` dan `AmazonS3ReadOnlyAccess` untuk menyertakan izin untuk S3 Lambda Objek.	27 September 2021
Amazon S3 menambahkan `AmazonS3ObjectLamb`	Amazon S3 menambahkan kebijakan AWS terkelola	18 Agustus 2021

Perubahan	Deskripsi	Tanggal
`daExecutionRolePolicy`	baru yang disebut yang `AmazonS3ObjectLambdaExecutionRolePolicy` menyediakan izin fungsi Lambda untuk berinteraksi dengan Lambda Objek S3 dan menulis ke log. CloudWatch	
Amazon S3 mulai melakukan pelacakan perubahan	Amazon S3 mulai melacak perubahan untuk kebijakan yang AWS dikelola.	18 Agustus 2021

Menggunakan Peran Terkait Layanan untuk Lensa Penyimpanan Amazon S3

Untuk menggunakan Lensa Penyimpanan Amazon S3 untuk mengumpulkan dan mengumpulkan metrik di semua akun Anda di AWS Organizations, Anda harus terlebih dahulu memastikan bahwa Lensa Penyimpanan S3 memiliki akses tepercaya yang diaktifkan oleh akun kelola di organisasi Anda. S3 Storage Lens membuat peran terkait layanan (SLR) untuk memungkinkannya mendapatkan daftar Akun AWS milik organisasi Anda. Daftar akun ini digunakan oleh Lensa Penyimpanan S3 untuk mengumpulkan metrik untuk sumber daya S3 di semua akun anggota ketika dasbor Lensa Penyimpanan S3 atau konfigurasi dibuat atau diperbarui.

Amazon S3 Storage Lens menggunakan peran terkait AWS Identity and Access Management layanan (IAM). Peran terkait layanan adalah tipe peran IAM unik yang ditautkan langsung ke Lensa Penyimpanan S3. Peran terkait layanan telah ditentukan sebelumnya oleh S3 Storage Lens dan mencakup semua izin yang diperlukan layanan untuk memanggil orang lain atas nama Anda. Layanan AWS

Peran terkait layanan membuat pengaturan Lensa Penyimpanan S3 menjadi lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Lensa Penyimpanan S3 menentukan izin peran terkait layanan, kecuali jika ditentukan berbeda, hanya Lensa Penyimpanan S3 yang dapat mengasumsikan peran tersebut. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, dan kebijakan izin tersebut tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran terkait layanan hanya setelah menghapus sumber daya terkait terlebih dahulu. Ini melindungi sumber daya Lensa Penyimpanan S3 Anda karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran terkait layanan, lihat Layanan AWS yang berfungsi dengan IAM dan cari layanan yang memiliki Ya di dalam Peran Terkait Layanan. Pilih Ya bersama tautan untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

Menggunakan peran terkait layanan untuk Lensa Penyimpanan Amazon S3

Lensa Penyimpanan S3 menggunakan peran terkait layanan bernama AWSServiceRoleForS3StorageLens— Ini memungkinkan akses ke AWS layanan dan sumber daya yang digunakan atau dikelola oleh S3 Storage Lens. Hal ini memungkinkan S3 Storage Lens untuk mengakses AWS Organizations sumber daya atas nama Anda.

Peran yang berkaitan dengan layanan Lensa Penyimpanan S3 memercayakan layanan berikut di penyimpanan organisasi Anda:

- `storage-lens.s3.amazonaws.com`

Kebijakan izin peran mengizinkan Lensa Penyimpanan S3 untuk menyelesaikan tindakan berikut:

- `organizations:DescribeOrganization`

  `organizations:ListAccounts`

  `organizations:ListAWSServiceAccessForOrganization`

  `organizations:ListDelegatedAdministrators`

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, lihat Izin peran tertaut layanan dalam Panduan Pengguna IAM.

Membuat peran terkait layanan untuk Lensa Penyimpanan S3

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda menyelesaikan salah satu tugas berikut saat masuk ke AWS Organizations manajemen atau akun administrator delegasi, S3 Storage Lens akan membuat peran terkait layanan untuk Anda:

- Buat konfigurasi dasbor Lensa Penyimpanan S3 untuk organisasi Anda di konsol Amazon S3.

- PUTkonfigurasi Lensa Penyimpanan S3 untuk organisasi Anda menggunakan REST API, AWS CLI dan SDK.

> **Note**
>
> S3 Storage Lens akan mendukung maksimal lima administrator yang didelegasikan per organisasi.

Jika Anda menghapus peran terkait layanan ini, tindakan sebelumnya akan dibuat ulang sesuai kebutuhan.

Kebijakan contoh untuk peran terkait layanan Lensa Penyimpanan S3

Example Kebijakan izin untuk peran terkait layanan Lensa Penyimpanan Amazon S3

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AwsOrgsAccess",
            "Effect": "Allow",
            "Action": [
                "organizations:DescribeOrganization",
                "organizations:ListAccounts",
                "organizations:ListAWSServiceAccessForOrganization",
                "organizations:ListDelegatedAdministrators"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Mengedit peran terkait layanan untuk Lensa Penyimpanan Amazon S3

S3 Storage Lens tidak memungkinkan Anda mengedit peran AWSServiceRoleForS3StorageLens terkait layanan. Setelah membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat menyunting

penjelasan peran menggunakan IAM. Untuk informasi selengkapnya, lihat Mengedit peran tertaut layanan dalam Panduan Pengguna IAM.

Menghapus peran terkait layanan untuk Lensa Penyimpanan Amazon S3

Jika Anda tidak perlu lagi menggunakan peran terkait layanan, kami rekomendasikan agar Anda menghapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas yang tidak digunakan yang tidak dipantau atau dipelihara secara aktif. Tetapi, Anda harus membersihkan sumber daya peran terkait layanan sebelum menghapusnya secara manual.

> **Note**
>
> Jika layanan Lensa Penyimpanan Amazon S3 menggunakan peran tersebut ketika Anda mencoba menghapus sumber daya, penghapusan mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk menghapus AWSServiceRoleForS3StorageLens Anda harus menghapus semua konfigurasi Lensa Penyimpanan S3 tingkat organisasi yang ada di semua Wilayah AWS menggunakan AWS Organizations manajemen atau akun administrator delegasi.

Sumber daya adalah konfigurasi Lensa Penyimpanan S3 tingkat organisasi. Gunakan S3 Storage Lens untuk membersihkan sumber daya dan kemudian gunakan Konsol IAM, CLI, REST API, atau AWS SDK untuk menghapus peran.

Di REST API, AWS CLI, dan SDK, konfigurasi Lensa Penyimpanan S3 dapat ditemukan menggunakan `ListStorageLensConfigurations` di semua Wilayah tempat organisasi Anda membuat konfigurasi Lensa Penyimpanan S3. Gunakan tindakan `DeleteStorageLensConfiguration` untuk menghapus konfigurasi ini sehingga Anda dapat menghapus peran.

> **Note**
>
> Untuk menghapus peran terkait layanan, Anda harus menghapus semua konfigurasi Lensa Penyimpanan S3 tingkat organisasi di semua Wilayah tempat mereka ada.

Untuk menghapus sumber daya Lensa Penyimpanan Amazon S3 yang digunakan oleh SLR AWSServiceRoleForS3StorageLens

1. Untuk mendapatkan daftar konfigurasi tingkat organisasi Anda, Anda harus menggunakan `ListStorageLensConfigurations` di setiap Wilayah yang memiliki konfigurasi Lensa Penyimpanan S3. Daftar ini juga dapat diperoleh dari konsol Amazon S3.
2. Hapus konfigurasi ini dari titik akhir Regional yang sesuai dengan menjalankan panggilan `DeleteStorageLensConfiguration` API atau dengan menggunakan konsol Amazon S3.

Untuk menghapus peran terkait layanan secara manual menggunakan IAM

Setelah Anda menghapus konfigurasi, hapus AWSServiceRoleForS3StorageLens SLR dari Konsol IAM atau dengan menjalankan API IAM`DeleteServiceLinkedRole`, atau menggunakan atau SDK. AWS CLI AWS Untuk informasi selengkapnya, lihat Menghapus peran tertaut layanan dalam Panduan Pengguna IAM.

Wilayah yang Di Support untuk Peran Terkait Layanan Lensa Penyimpanan S3

S3 Storage Lens mendukung penggunaan peran terkait layanan di semua Wilayah AWS tempat layanan tersedia. Untuk informasi lebih lanjut, lihat Wilayah dan Titik Akhir Amazon S3.

Memecahkan masalah identitas dan akses Amazon S3

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan Amazon S3 dan IAM.

Topik

- Saya menerima kesalahan akses ditolak
- Saya tidak berwenang untuk melakukan tindakan di Amazon S3
- Saya tidak berwenang untuk melakukan iam: PassRole
- Saya ingin mengizinkan orang di luar saya Akun AWS mengakses sumber daya Amazon S3 saya

Saya menerima kesalahan akses ditolak

Verifikasi bahwa tidak ada `Deny` pernyataan eksplisit terhadap pemohon yang Anda coba berikan izin baik dalam kebijakan bucket atau kebijakan berbasis identitas.

Untuk informasi rinci tentang pemecahan masalah akses ditolak kesalahan, lihat. Pecahkan masalah kesalahan Akses Ditolak (403 Forbidden) di Amazon S3

Saya tidak berwenang untuk melakukan tindakan di Amazon S3

Jika Anda menerima pesan kesalahan bahwa Anda tidak memiliki otorisasi untuk melakukan tindakan, kebijakan Anda harus diperbarui agar Anda dapat melakukan tindakan tersebut.

Contoh kesalahan berikut terjadi ketika pengguna IAM `mateojackson` mencoba menggunakan konsol untuk melihat detail tentang suatu sumber daya *`my-example-widget`* rekaan, tetapi tidak memiliki izin `s3:`*`GetWidget`* rekaan.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 s3:GetWidget on resource: my-example-widget
```

Dalam hal ini, kebijakan untuk pengguna `mateojackson` harus diperbarui untuk mengizinkan akses ke sumber daya *`my-example-widget`* dengan menggunakan tindakan `s3:`*`GetWidget`*.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya tidak berwenang untuk melakukan iam: PassRole

Jika Anda menerima kesalahan bahwa Anda tidak diizinkan untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran ke Amazon S3.

Beberapa Layanan AWS memungkinkan Anda untuk meneruskan peran yang ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait layanan. Untuk melakukannya, Anda harus memiliki izin untuk meneruskan peran ke layanan.

Contoh kesalahan berikut terjadi ketika pengguna IAM bernama `marymajor` mencoba menggunakan konsol untuk melakukan tindakan di Amazon S3. Namun, tindakan tersebut memerlukan layanan untuk mendapatkan izin yang diberikan oleh peran layanan. Mary tidak memiliki izin untuk meneruskan peran tersebut pada layanan.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dalam kasus ini, kebijakan Mary harus diperbarui agar dia mendapatkan izin untuk melakukan tindakan `iam:PassRole` tersebut.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya ingin mengizinkan orang di luar saya Akun AWS mengakses sumber daya Amazon S3 saya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau orang-orang di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa saja yang dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACL), Anda dapat menggunakan kebijakan tersebut untuk memberi orang akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa referensi berikut:

- Untuk mempelajari apakah Amazon S3 mendukung fitur-fitur ini, lihat. Bagaimana Amazon S3 bekerja dengan IAM
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda di seluruh sumber daya Akun AWS yang Anda miliki, lihat Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki di Panduan Pengguna IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda kepada pihak ketiga Akun AWS, lihat Menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat Menyediakan akses ke pengguna terautentikasi eksternal (federasi identitas) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara menggunakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM.

Mengelola akses dengan S3 Access Grants

Untuk mematuhi prinsip hak akses paling rendah, Anda menentukan akses terperinci ke data Amazon S3 Anda berdasarkan aplikasi, persona, grup, atau unit organisasi. Anda dapat menggunakan berbagai pendekatan untuk mencapai akses terperinci ke data Anda di Amazon S3, tergantung pada skala dan kompleksitas pola akses.

Pendekatan paling sederhana untuk mengelola akses ke small-to-medium jumlah kumpulan data di Amazon S3 AWS Identity and Access Management by (IAM) prinsipal adalah dengan menentukan kebijakan izin IAM dan kebijakan bucket S3. Strategi ini berhasil, selama kebijakan yang diperlukan sesuai dengan batas ukuran kebijakan bucket S3 (20 KB) dan kebijakan IAM (5 KB), dan dalam jumlah pengguna utama IAM yang diizinkan per akun.

Karena jumlah set data dan kasus penggunaan Anda meningkat, Anda mungkin memerlukan lebih banyak ruang kebijakan. Pendekatan yang menawarkan lebih banyak ruang secara signifikan

untuk pernyataan kebijakan adalah dengan menggunakan Titik Akses S3 sebagai titik akhir tambahan untuk bucket S3, karena setiap titik akses dapat memiliki kebijakannya sendiri. Anda dapat menentukan pola kontrol akses yang cukup terperinci, karena Anda dapat memiliki ribuan titik akses Wilayah AWS per akun, dengan kebijakan hingga 20 KB untuk setiap titik akses. Meskipun S3 Access Points meningkatkan jumlah ruang kebijakan yang tersedia, diperlukan mekanisme bagi klien untuk menemukan titik akses yang tepat untuk set data yang tepat.

Pendekatan ketiga adalah menerapkan pola broker sesi IAM, di mana Anda menerapkan logika keputusan akses dan secara dinamis menghasilkan kredensial sesi IAM jangka pendek untuk setiap sesi akses. Sementara pendekatan broker sesi IAM mendukung pola dan skala izin dinamis yang sewenang-wenang secara efektif, Anda harus membangun logika pola akses.

Alih-alih menggunakan pendekatan ini, Anda dapat menggunakan S3 Access Grants untuk mengelola akses ke data Amazon S3. S3 Access Grants menyediakan model yang disederhanakan untuk menentukan izin akses ke data di Amazon S3 berdasarkan prefiks, bucket, atau objek. Selain itu, Anda dapat menggunakan S3 Access Grants untuk memberikan akses ke kedua pengguna utama IAM dan langsung ke pengguna atau grup dari direktori perusahaan Anda.

Anda biasanya menentukan izin untuk data di Amazon S3 dengan memetakan pengguna dan grup ke set data. Anda dapat menggunakan S3 Access Grants untuk menentukan pemetaan akses langsung prefiks S3 ke pengguna dan peran dalam bucket dan objek Amazon S3. Dengan skema akses yang disederhanakan di S3 Access Grants, Anda dapat memberikan akses read-only, write-only, atau read-write berdasarkan prefiks per-S3 ke kedua pengguna utama IAM dan langsung ke pengguna atau grup dari direktori perusahaan. Dengan kemampuan S3 Access Grants ini, aplikasi dapat meminta data dari Amazon S3 atas nama pengguna aplikasi yang diautentikasi saat ini.

Saat Anda mengintegrasikan S3 Access Grants dengan fitur propagasi identitas tepercaya AWS IAM Identity Center, aplikasi Anda dapat mengajukan permintaan ke Layanan AWS (termasuk S3 Access Grants) secara langsung atas nama pengguna direktori perusahaan yang diautentikasi. Aplikasi Anda tidak perlu lagi memetakan pengguna terlebih dahulu ke pengguna utama IAM. Selain itu, karena identitas pengguna akhir disebarkan sampai ke Amazon S3, mengaudit pengguna mana yang mengakses S3 Object mana yang disederhanakan. Anda tidak perlu lagi merekonstruksi hubungan antara pengguna yang berbeda dan sesi IAM. Saat Anda menggunakan S3 Access Grants dengan propagasi identitas tepercaya IAM Identity Center, setiap AWS CloudTrailperistiwa data untuk Amazon S3 berisi referensi langsung ke pengguna akhir atas nama siapa data tersebut diakses.

Untuk informasi selengkapnya tentang S3 Access Grants, lihat topik-topik berikut.

Topik

- Konsep S3 Access Grants
- S3 Access Grants dan identitas direktori perusahaan
- Memulai S3 Access Grants
- Membuat instans S3 Access Grants
- Mendaftarkan lokasi
- Membuat pemberian
- Minta akses ke data Amazon S3 melalui S3 Access Grants
- Akses data S3 melalui hibah akses
- Akses S3 memberikan akses lintas akun
- Menggunakan AWS tag dengan S3 Access Grants
- Batasan S3 Access Grants
- Integrasi S3 Access Grants

Konsep S3 Access Grants

S3 Access Grants memperkenalkan konsep-konsep berikut untuk skema aksesnya yang disederhanakan:

Instans S3 Access Grants

Instans S3 Access Grants adalah kontainer logis untuk masing-masing pemberian yang menentukan siapa yang memiliki tingkat akses apa ke data Amazon S3 apa. Anda dapat memiliki satu instans S3 Access Grants per Wilayah AWS per Akun AWS. Anda menggunakan instance S3 Access Grants ini untuk mengontrol akses ke semua bucket di akun yang sama dan. Wilayah AWS Jika Anda ingin menggunakan S3 Access Grants untuk memberikan akses ke identitas pengguna dan grup di direktori perusahaan Anda, Anda juga harus mengaitkan instance S3 Access Grants Anda dengan instans Pusat Identitas AWS Identity and Access Management (IAM).

Lokasi

Lokasi menentukan data mana yang dapat diberikan akses oleh instans S3 Access Grants Anda. S3 Access Grants bekerja dengan menjual kredensial IAM dengan akses tercakup ke prefiks, bucket, atau S3 Object tertentu. Anda mengaitkan lokasi S3 Access Grants dengan peran IAM, dari mana sesi sementara ini dibuat. Konfigurasi lokasi yang paling umum adalah satu lokasi di `s3://` untuk seluruh instans S3 Access Grants, yang dapat mencakup akses ke semua bucket

S3 di akun dan Wilayah AWS. Anda juga dapat membuat beberapa lokasi di instans S3 Access Grants. Misalnya, Anda dapat mendaftarkan bucket sebagai lokasi `s3://example-s3-bucket1` hibah yang ingin Anda batasi ke bucket ini, dan Anda juga dapat mendaftarkan lokasi `s3://` default.

Izin

Untuk mempersempit ruang lingkup akses dalam suatu lokasi, Anda membuat pemberian individual. Pemberian individu dalam instans S3 Access Grants memungkinkan entitas tertentu —pengguna utama IAM, atau pengguna atau grup di direktori perusahaan—mengakses prefiks, bucket, atau objek Amazon S3. Untuk setiap pemberian, Anda dapat menentukan cakupan yang berbeda (prefiks, bucket, atau objek) dan tingkat akses (`READ`, `WRITE`, atau `READWRITE`). Misalnya, Anda mungkin memiliki pemberian yang memungkinkan grup direktori perusahaan tertentu, `01234567-89ab-cdef-0123-456789abcdef` `READ` akses ke `s3://example-s3-bucket1/projects/items/*`. Hibah ini memberikan pengguna di grup tersebut `READ` akses ke setiap objek yang memiliki nama kunci dengan prefiks `projects/items/` di bucket bernama `example-s3-bucket1`.

Akses S3 memberikan kredensial sementara

Aplikasi dapat meminta kredensyal just-in-time akses dengan memanggil operasi API S3 baru, GetDataAccess, untuk meminta akses ke satu objek, awalan, atau bucket dengan tingkat izin,, atau. `READ` `WRITE` `READWRITE` Instans S3 Access Grants mengevaluasi permintaan `GetDataAccess` terhadap pemberian yang dimilikinya. Jika ada pemberian yang cocok, S3 Access Grants mengasumsikan peran IAM yang terkait dengan lokasi pemberian yang cocok. S3 Access Grants kemudian mencakup izin sesi IAM ke bucket, prefiks, atau S3 Object yang ditentukan oleh cakupan pemberian. Waktu kedaluwarsa kredensyal akses sementara default menjadi 1 jam, tetapi Anda dapat mengaturnya ke nilai apa pun dari 15 menit hingga 12 jam.

Cara kerjanya

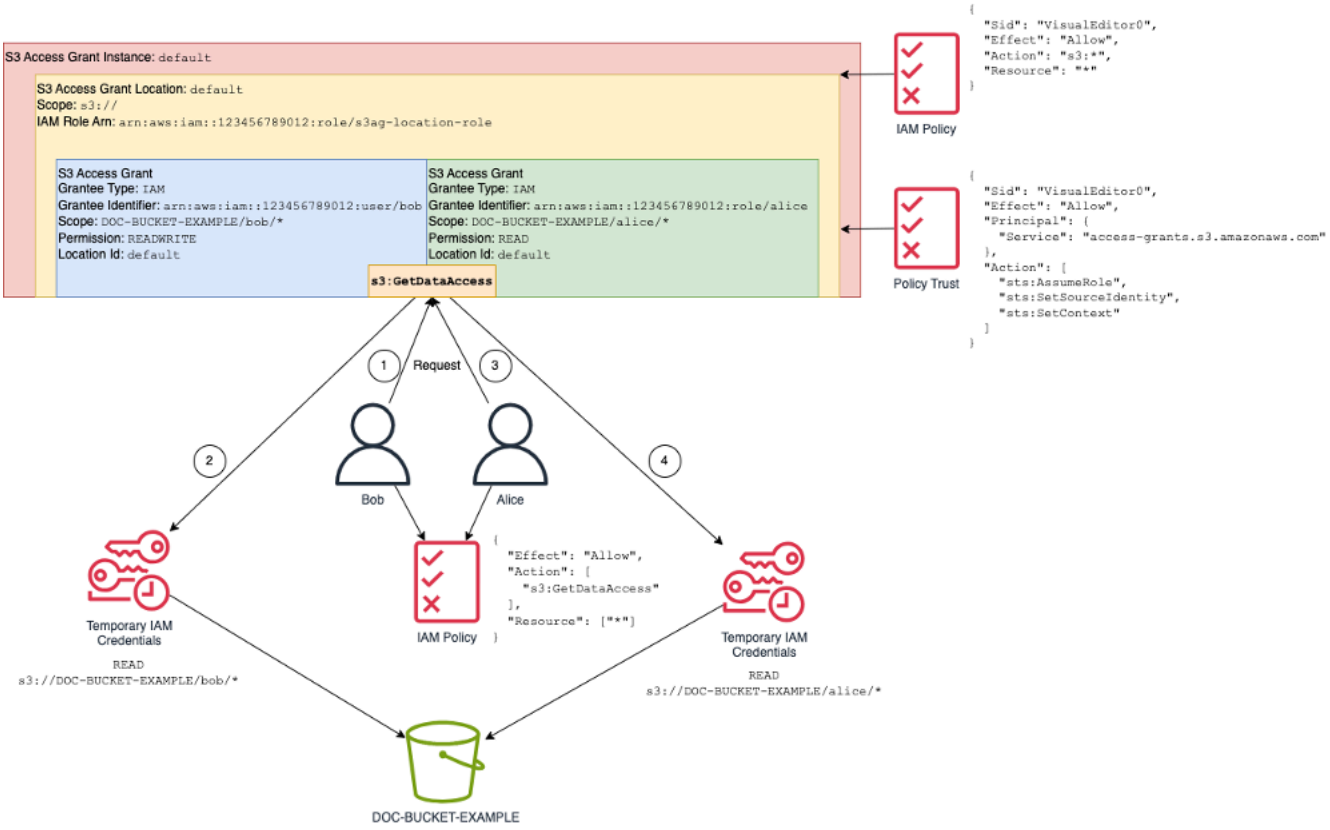
Dalam diagram berikut, lokasi Amazon S3 default dengan cakupan `s3://` terdaftar dengan peran IAM `s3ag-location-role`. Peran IAM ini memiliki izin untuk melakukan tindakan Amazon S3 dalam akun saat kredensialnya diperoleh melalui S3 Access Grants.

Di lokasi ini, dua pemberian akses individu dibuat untuk dua pengguna IAM. Pengguna IAM Bob diberikan keduanya `READ` dan `WRITE` akses pada `bob/` prefiks di bucket `DOC-BUCKET-EXAMPLE`. Peran IAM lainnya, Alice, hanya diberikan `READ` akses pada `alice/` awalan di ember. `DOC-BUCKET-EXAMPLE` Hibah, berwarna biru, didefinisikan untuk Bob untuk mengakses prefiks `bob/` di

DOC-BUCKET-EXAMPLE bucket. Hibah, berwarna hijau, didefinisikan untuk Alice untuk mengakses prefiks alice/ di DOC-BUCKET-EXAMPLE bucket.

Saat tiba waktunya bagi Bob untuk melakukan READ data, peran IAM yang terkait dengan lokasi hibahnya memanggil operasi S3 Access Grants API GetDataAccess. Jika Bob mencoba READ prefiks atau S3 Object yang dimulai dengans3://DOC-BUCKET-EXAMPLE/bob/*, GetDataAccess permintaan mengembalikan satu set kredensial sesi IAM sementara dengan izin untuk s3://DOC-BUCKET-EXAMPLE/bob/*. Demikian pula, Bob dapat WRITE ke prefiks atau S3 Object apa pun yang dimulai dengan s3://DOC-BUCKET-EXAMPLE/bob/*, karena pemberian juga memungkinkan itu.

Demikian pula, Alice bisa READ apa saja yang dimulai dengan s3://DOC-BUCKET-EXAMPLE/alice/. Namun, jika dia mencoba WRITE apa pun ke bucket, prefiks, atau objek apa pun s3://, dia akan mendapatkan kesalahan Access Denied (403 Forbidden), karena tidak ada pemberian yang memberikannya WRITE akses ke data apa pun. Selain itu, jika Alice meminta tingkat akses apa pun (READ atau WRITE) ke data di luar s3://DOC-BUCKET-EXAMPLE/alice/, dia akan kembali menerima kesalahan Akses Ditolak.



Pola ini menskalakan ke sejumlah besar pengguna dan bucket dan menyederhanakan pengelolaan izin tersebut. Daripada mengedit kebijakan bucket S3 yang berpotensi besar setiap kali Anda

ingin menambahkan atau menghapus hubungan akses prefiks pengguna individual, Anda dapat menambahkan dan menghapus masing-masing pemberian terpisah.

S3 Access Grants dan identitas direktori perusahaan

Anda dapat menggunakan Amazon S3 Access Grants untuk memberikan akses ke prinsipal AWS Identity and Access Management (IAM) (pengguna atau peran), baik dalam hal yang sama maupun yang lain. Akun AWS Namun, dalam banyak kasus, entitas yang mengakses data adalah pengguna akhir dari direktori perusahaan Anda. Alih-alih memberikan akses ke pengguna utama IAM, Anda dapat menggunakan S3 Access Grants untuk memberikan akses langsung ke pengguna dan grup perusahaan Anda. Dengan S3 Access Grants, Anda tidak perlu lagi memetakan identitas perusahaan Anda ke pengguna utama IAM menengah untuk mengakses data S3 Anda melalui aplikasi perusahaan Anda.

Fungsionalitas baru ini—dukungan untuk menggunakan akses identitas pengguna akhir ke data—disediakan dengan mengaitkan instance S3 Access Grants Anda dengan sebuah instance. AWS IAM Identity Center IAM Identity Center mendukung penyedia identitas berbasis standar dan merupakan hub AWS untuk layanan atau fitur apa pun, termasuk S3 Access Grants, yang mendukung identitas pengguna akhir. IAM Identity Center menyediakan dukungan otentikasi untuk identitas perusahaan melalui fitur propagasi identitas tepercaya. Untuk informasi selengkapnya, lihat Propagasi identitas terpercaya di seluruh aplikasi.

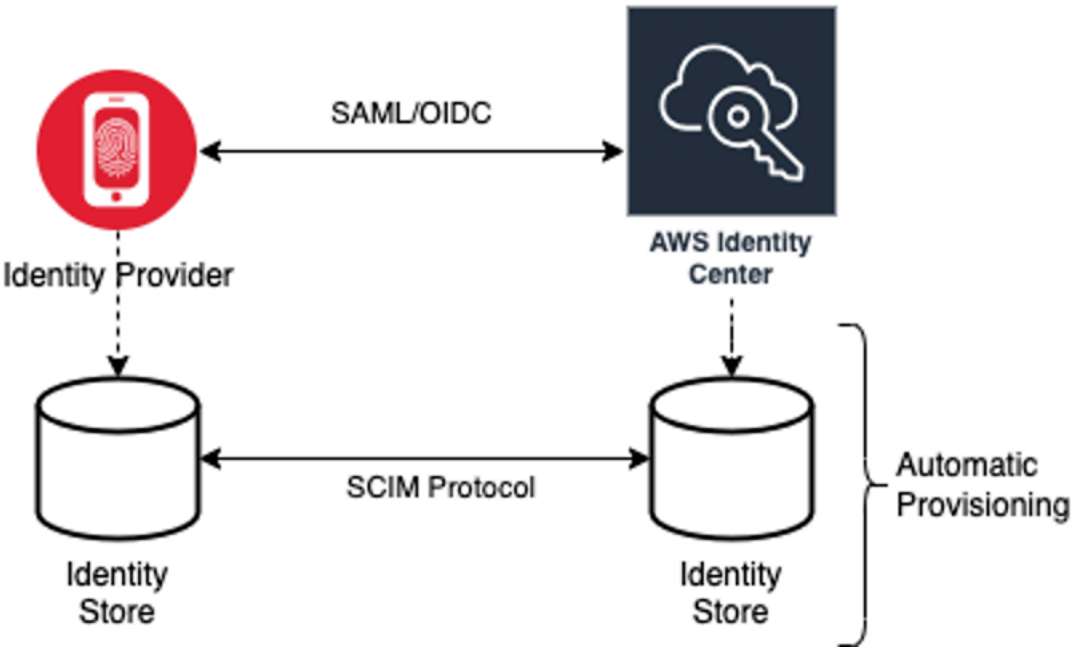
Untuk memulai dengan dukungan identitas tenaga kerja di S3 Access Grants, sebagai prasyarat, Anda mulai di IAM Identity Center dengan mengonfigurasi penyediaan identitas antara penyedia identitas perusahaan Anda dan IAM Identity Center. IAM Identity Center mendukung penyedia identitas perusahaan seperti Okta, Microsoft Entra ID (sebelumnya Azure Active Directory), atau penyedia identitas (IdP) eksternal lainnya yang mendukung protokol System for Cross-domain Identity Management (SCIM). Saat Anda menghubungkan Pusat Identitas IAM ke IdP dan mengaktifkan penyediaan otomatis, pengguna dan grup dari IdP Anda disinkronkan ke dalam penyimpanan identitas di Pusat Identitas IAM. Setelah langkah ini, IAM Identity Center memiliki pandangan sendiri tentang pengguna dan grup Anda, sehingga Anda dapat merujuk kepada mereka dengan menggunakan fitur lain Layanan AWS dan fitur, seperti S3 Access Grants. Untuk informasi selengkapnya tentang mengonfigurasi penyediaan otomatis Pusat Identitas IAM, lihat Penyediaan otomatis di Panduan Pengguna AWS IAM Identity Center .

IAM Identity Center terintegrasi AWS Organizations sehingga Anda dapat mengelola izin secara terpusat di beberapa Akun AWS tanpa mengonfigurasi setiap akun secara manual. Dalam organisasi tipikal, administrator identitas Anda mengonfigurasi satu instans Pusat Identitas IAM untuk seluruh organisasi, sebagai satu titik sinkronisasi identitas. Instance IAM Identity Center ini biasanya berjalan

di dedicated Akun AWS di organisasi Anda. Dalam konfigurasi umum ini, Anda dapat merujuk ke identitas pengguna dan grup di S3 Access Grants dari manapun Akun AWS di organisasi.

Namun, jika AWS Organizations administrator belum mengonfigurasi instans Pusat Identitas IAM pusat, Anda dapat membuat instance lokal di akun yang sama dengan instans S3 Access Grants Anda. Konfigurasi seperti itu lebih umum untuk proof-of-concept atau kasus penggunaan pengembangan lokal. Dalam semua kasus, instance IAM Identity Center harus Wilayah AWS sama dengan instance S3 Access Grants yang akan dikaitkan.

Dalam diagram berikut konfigurasi Pusat Identitas IAM dengan IdP eksternal, IdP dikonfigurasi dengan SCIM untuk menyinkronkan penyimpanan identitas dari IdP ke penyimpanan identitas di IAM Identity Center.



Untuk menggunakan identitas direktori perusahaan Anda dengan S3 Access Grants, lakukan hal berikut ini:

- Siapkan penyediaan Otomatis di Pusat Identitas IAM untuk menyinkronkan informasi pengguna dan grup dari IdP Anda ke Pusat Identitas IAM.
- Konfigurasikan sumber identitas eksternal Anda dalam IAM Identity Center sebagai penerbit token tepercaya. Untuk informasi selengkapnya, lihat Propagasi identitas tepercaya di seluruh aplikasi di Panduan AWS IAM Identity Center Pengguna.
- Kaitkan instans S3 Access Grants Anda dengan instans Pusat Identitas IAM Anda. Anda dapat melakukan ini saat membuat instans S3 Access Grants Anda. Jika Anda telah membuat instance S3 Access Grants, lihat. Kaitkan atau pisahkan instans Pusat Identitas IAM Anda

Bagaimana identitas direktori dapat mengakses data S3

Misalkan Anda memiliki pengguna direktori perusahaan yang perlu mengakses data S3 Anda melalui aplikasi perusahaan, misalnya, aplikasi penampil dokumen, yang terintegrasi dengan IdP eksternal Anda (misalnya, Okta) untuk mengautentikasi pengguna. Otentikasi pengguna dalam aplikasi ini biasanya dilakukan melalui pengalihan di browser web pengguna. Karena pengguna dalam direktori bukan pengguna utama IAM, aplikasi Anda memerlukan kredensial IAM yang dapat memanggil operasi S3 Access Grants `GetDataAccess` API untuk mendapatkan kredensial akses ke data S3 atas nama pengguna. Tidak seperti pengguna IAM dan peran yang mendapatkan kredensialnya sendiri, aplikasi Anda memerlukan cara untuk mewakili pengguna direktori, yang tidak dipetakan ke peran IAM, sehingga pengguna bisa mendapatkan akses data melalui S3 Access Grants.

Transisi ini, dari pengguna direktori yang diautentikasi ke penelepon IAM yang dapat membuat permintaan ke S3 Access Grants atas nama pengguna direktori, dilakukan oleh aplikasi melalui fitur penerbit token tepercaya IAM Identity Center. Aplikasi, setelah mengautentikasi pengguna direktori, memiliki token identitas dari IDP (misalnya, Okta) yang mewakili pengguna direktori menurut Okta. Konfigurasi penerbit token tepercaya di IAM Identity Center memungkinkan aplikasi untuk menukar Okta token ini (Oktapenyewa dikonfigurasi sebagai “penerbit tepercaya”) untuk token identitas yang berbeda dari IAM Identity Center yang akan mewakili pengguna direktori dengan aman Layanan AWS. Aplikasi data kemudian akan mengambil peran IAM, menyediakan token pengguna direktori dari IAM Identity Center sebagai konteks tambahan. Aplikasi dapat menggunakan sesi IAM yang dihasilkan untuk memanggil S3 Access Grants. Token mewakili identitas aplikasi (pengguna utama IAM itu sendiri) serta identitas pengguna direktori.

Langkah utama dari transisi ini adalah pertukaran token. Aplikasi melakukan pertukaran token ini dengan memanggil operasi `CreateTokenWithIAM` API di IAM Identity Center. Tentu saja, itu juga merupakan panggilan AWS API dan membutuhkan kepala sekolah IAM untuk menandatanganinya. Pengguna utama IAM yang membuat permintaan ini biasanya merupakan peran IAM yang terkait dengan aplikasi. Misalnya, jika aplikasi berjalan di Amazon EC2, permintaan `CreateTokenWithIAM` biasanya dilakukan oleh peran IAM yang terkait dengan instans EC2 tempat aplikasinya dijalankan. Hasil dari `CreateTokenWithIAM` panggilan yang berhasil adalah token identitas baru, yang akan dikenali di dalamnya Layanan AWS.

Langkah selanjutnya, sebelum aplikasi dapat memanggil `GetDataAccess` nama pengguna direktori, adalah agar aplikasi mendapatkan sesi IAM yang mencakup identitas pengguna direktori. Aplikasi melakukan ini dengan `AssumeRole` permintaan AWS Security Token Service (AWS STS) yang juga menyertakan token IAM Identity Center untuk pengguna direktori sebagai konteks identitas tambahan. Konteks tambahan inilah yang memungkinkan IAM Identity Center untuk menyebarkan

identitas pengguna direktori ke langkah berikutnya. Peran IAM yang diasumsikan aplikasi adalah peran yang memerlukan izin IAM untuk memanggil operasi `GetDataAccess`.

Setelah mengasumsikan peran IAM pembawa identitas dengan token IAM Identity Center untuk pengguna direktori sebagai konteks tambahan, aplikasi sekarang memiliki semua yang diperlukan untuk membuat permintaan yang ditandatangani `GetDataAccess` atas nama pengguna direktori yang diautentikasi.

Penyebaran token didasarkan pada langkah-langkah berikut:

Buat aplikasi Pusat Identitas IAM

Pertama, buat aplikasi baru di IAM Identity Center. Aplikasi ini akan menggunakan template yang memungkinkan IAM Identity Center untuk mengidentifikasi jenis pengaturan aplikasi yang dapat Anda gunakan. Perintah untuk membuat aplikasi mengharuskan Anda untuk memberikan contoh IAM Identity Center Amazon Resource Name (ARN), nama aplikasi, dan penyedia aplikasi ARN. Penyedia aplikasi adalah penyedia aplikasi SAMB atau OAuth yang akan digunakan aplikasi untuk melakukan panggilan ke IAM Identity Center.

Untuk menggunakan perintah contoh berikut, ganti *`user input placeholders`* dengan informasi Anda sendiri:

```
aws sso-admin create-application \
 --instance-arn "arn:aws:sso:::instance/ssoins-ssoins-1234567890abcdef" \
 --application-provider-arn "arn:aws:sso::aws:applicationProvider/custom" \
 --name MyDataApplication
```

Respons:

```
{
   "ApplicationArn": "arn:aws:sso::123456789012:application/ssoins-
ssoins-1234567890abcdef/apl-abcd1234a1b2c3d"
}
```

Buat penerbit token tepercaya

Sekarang setelah Anda memiliki aplikasi IAM Identity Center, langkah selanjutnya adalah mengonfigurasi penerbit token tepercaya yang akan digunakan untuk menukar nilai `IdToken` Anda dari IdP Anda dengan token IAM Identity Center. Pada langkah ini Anda perlu memberikan item berikut:

- URL penerbit penyedia identitas
- Nama penerbit token tepercaya
- Jalur atribut klaim
- Jalur atribut toko identitas
- Opsi pengambilan JSON Web Key Set (JWKS)

Jalur atribut klaim adalah atribut penyedia identitas yang akan digunakan untuk memetakan ke atribut penyimpanan identitas. Biasanya, jalur atribut klaim adalah alamat email pengguna, tetapi Anda dapat menggunakan atribut lain untuk melakukan pemetaan.

Buat file yang disebut `oidc-configuration.json` dengan informasi berikut. Untuk menggunakan file ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
{
  "OidcJwtConfiguration":
     {
      "IssuerUrl": "https://login.microsoftonline.com/a1b2c3d4-abcd-1234-b7d5-
b154440ac123/v2.0",
      "ClaimAttributePath": "preferred_username",
      "IdentityStoreAttributePath": "userName",
      "JwksRetrievalOption": "OPEN_ID_DISCOVERY"
     }
}
```

Untuk membuat penerbit token tepercaya, jalankan perintah berikut. Untuk menggunakan contoh perintah ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
aws sso-admin create-trusted-token-issuer \
  --instance-arn "arn:aws:sso:::instance/ssoins-1234567890abcdef" \
  --name MyEntraIDTrustedIssuer \
  --trusted-token-issuer-type OIDC_JWT \
  --trusted-token-issuer-configuration file://./oidc-configuration.json
```

Respons

```
{
  "TrustedTokenIssuerArn": "arn:aws:sso::123456789012:trustedTokenIssuer/
ssoins-1234567890abcdef/tti-43b4a822-1234-1234-1234-a1b2c3d41234"
}
```

Connect aplikasi IAM Identity Center dengan penerbit token tepercaya

Penerbit token tepercaya memerlukan beberapa pengaturan konfigurasi lagi untuk bekerja. Tetapkan audiens yang akan dipercaya oleh penerbit token tepercaya. Audiens adalah nilai di dalam `IdToken` yang diidentifikasi oleh kunci dan dapat ditemukan di pengaturan penyedia identitas. Sebagai contoh:

```
1234973b-abcd-1234-abcd-345c5a9c1234
```

Buat file dengan nama `grant.json` dengan konten berikut ini. Untuk menggunakan file ini, ubah audiens agar sesuai dengan pengaturan penyedia identitas Anda dan berikan ARN penerbit token tepercaya yang dikembalikan oleh perintah sebelumnya.

```
{
   "JwtBearer":
     {
       "AuthorizedTokenIssuers":
         [
           {
             "TrustedTokenIssuerArn": "arn:aws:sso::123456789012:trustedTokenIssuer/
ssoins-1234567890abcdef/tti-43b4a822-1234-1234-1234-a1b2c3d41234",
               "AuthorizedAudiences":
                 [
                   "1234973b-abcd-1234-abcd-345c5a9c1234"
                 ]
            }
         ]
     }
 }
```

Jalankan perintah contoh berikut. Untuk menggunakan perintah ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
aws sso-admin put-application-grant \
  --application-arn "arn:aws:sso::123456789012:application/ssoins-
ssoins-1234567890abcdef/apl-abcd1234a1b2c3d" \
  --grant-type "urn:ietf:params:oauth:grant-type:jwt-bearer" \
  --grant file://./grant.json \
```

Perintah ini menetapkan penerbit token tepercaya dengan pengaturan konfigurasi untuk mempercayai audiens dalam file `grant.json` dan menautkan audiens ini dengan aplikasi yang dibuat pada langkah pertama untuk bertukar token dari jenis `jwt-bearer`. String

`urn:ietf:params:oauth:grant-type:jwt-bearer` bukan merupakan string arbitrer. Ini adalah namespace terdaftar di profil pernyataan OAuth JSON Web Token (JWT). Anda dapat menemukan informasi lebih lanjut tentang namespace ini di RFC 7523.

Selanjutnya, gunakan perintah berikut untuk mengatur cakupan mana yang akan disertakan oleh penerbit token tepercaya saat bertukar `IdToken` nilai dari penyedia identitas Anda. Untuk S3 Access Grants, nilai untuk `--scope` parameter adalah `s3:access_grants:read_write`.

```
aws sso-admin put-application-access-scope \
  --application-arn "arn:aws:sso::111122223333:application/ssoins-
ssoins-111122223333abcdef/apl-abcd1234a1b2c3d" \
  --scope "s3:access_grants:read_write"
```

Langkah terakhir adalah melampirkan kebijakan sumber daya ke aplikasi IAM Identity Center. Kebijakan ini akan memungkinkan peran IAM aplikasi Anda untuk membuat permintaan ke operasi API `sso-oauth:CreateTokenWithIAM` dan menerima `IdToken` nilai dari IAM Identity Center.

Buat file dengan nama `authentication-method.json` dengan konten berikut ini. Ganti *123456789012* dengan ID akun Anda.

```
{
   "Iam":
       {
         "ActorPolicy":
             {
                "Version": "2012-10-17",
                    "Statement":
                    [
                        {
                           "Effect": "Allow",
                            "Principal":
                            {
                              "AWS": "arn:aws:iam::123456789012:role/webapp"
                            },
                           "Action": "sso-oauth:CreateTokenWithIAM",
                            "Resource": "*"
                        }
                    ]
                }
            }
        }
```

Untuk melampirkan kebijakan ke aplikasi IAM Identity Center, jalankan perintah berikut:

```
aws sso-admin put-application-authentication-method \
   --application-arn "arn:aws:sso::123456789012:application/ssoins-
ssoins-1234567890abcdef/apl-abcd1234a1b2c3d" \
   --authentication-method-type IAM \
   --authentication-method file://./authentication-method.json
```

Ini melengkapi pengaturan konfigurasi untuk menggunakan S3 Access Grants dengan pengguna direktori melalui aplikasi web. Anda dapat menguji penyiapan ini secara langsung di aplikasi atau Anda dapat memanggil operasi `CreateTokenWithIAM` API dengan menggunakan perintah berikut dari peran IAM yang diizinkan dalam kebijakan aplikasi Pusat Identitas IAM:

```
aws sso-oidc create-token-with-iam \
   --client-id "arn:aws:sso::123456789012:application/ssoins-ssoins-1234567890abcdef/
apl-abcd1234a1b2c3d"  \
   --grant-type urn:ietf:params:oauth:grant-type:jwt-bearer \
   --assertion IdToken
```

Respons akan serupa dengan ini:

```
{
    "accessToken": "<suppressed long string to reduce space>",
    "tokenType": "Bearer",
    "expiresIn": 3600,
    "refreshToken": "<suppressed long string to reduce space>",
    "idToken": "<suppressed long string to reduce space>",
    "issuedTokenType": "urn:ietf:params:oauth:token-type:refresh_token",
    "scope": [
      "sts:identity_context",
      "s3:access_grants:read_write",
      "openid",
      "aws"
    ]
}
```

Jika Anda memecahkan kode `IdToken` nilai yang dikodekan dengan base64, Anda dapat melihat pasangan kunci-nilai dalam format JSON. Kunci `sts:identity_context` berisi nilai yang perlu dikirim aplikasi Anda dalam `sts:AssumeRole` permintaan untuk menyertakan informasi identitas pengguna direktori. Berikut adalah contoh yang `IdToken` di-decode:

```
{
    "aws:identity_store_id": "d-996773e796",
    "sts:identity_context": "AQoJb3JpZ2luX2VjEOTtl;<SUPRESSED>",
    "sub": "83d43802-00b1-7054-db02-f1d683aacba5",
    "aws:instance_account": "123456789012",
    "iss": "https://identitycenter.amazonaws.com/ssoins-1234567890abcdef",
    "sts:audit_context": "AQoJb3JpZ2luX2VjEOT<SUPRESSED>==",
    "aws:identity_store_arn": "arn:aws:identitystore::232642235904:identitystore/
d-996773e796",
    "aud": "abcd12344U0gi7n4Yyp0-WV1LWNlbnRyYWwtMQ",
    "aws:instance_arn": "arn:aws:sso:::instance/ssoins-6987d7fb04cf7a51",
    "aws:credential_id": "EXAMPLEHI5glPh40y9TpApJn8...",
    "act": {
       "sub": "arn:aws:sso::232642235904:trustedTokenIssuer/
ssoins-6987d7fb04cf7a51/43b4a822-1020-7053-3631-cb2d3e28d10e"
    },
    "auth_time": "2023-11-01T20:24:28Z",
    "exp": 1698873868,
    "iat": 1698870268
}
```

Anda bisa mendapatkan nilai dari `sts:identity_context` dan meneruskan informasi ini dalam `sts:AssumeRole` panggilan. Berikut ini adalah contoh CLI dari sintaks. Peran yang akan diasumsikan adalah peran sementara dengan izin untuk menginvokasi `s3:GetDataAccess`.

```
aws sts assume-role \
   --role-arn "arn:aws:iam::123456789012:role/temp-role" \
   --role-session-name "TempDirectoryUserRole" \
   --provided-contexts ProviderArn="arn:aws:iam::aws:contextProvider/
IdentityCenter",ContextAssertion="value from sts:identity_context"
```

Anda sekarang dapat menggunakan kredensial yang diterima dari panggilan ini untuk menginvokasi operasi `s3:GetDataAccess` API dan menerima kredensial akhir dengan akses ke sumber daya S3 Anda.

Memulai S3 Access Grants

Amazon S3 Access Grants adalah fitur Amazon S3 yang menyediakan solusi kontrol akses yang dapat diskalakan untuk data S3 Anda. S3 Access Grants adalah vendor kredensial S3, artinya Anda mendaftar dengan S3 Access Grants daftar pemberian Anda dan pada tingkat apa. Setelah itu, ketika pengguna atau klien membutuhkan akses ke data S3 Anda, mereka terlebih dahulu meminta S3

Access Grants untuk kredensial. Jika ada pemberian terkait yang mengotorisasi akses, S3 Access Grants menjual kredensial akses sementara dengan hak akses paling rendah. Pengguna atau klien kemudian dapat menggunakan kredensial penjual S3 Access Grants untuk mengakses data S3 Anda. Dengan mengingat hal itu, jika persyaratan data S3 Anda mengamanatkan konfigurasi izin yang kompleks atau besar, Anda dapat menggunakan S3 Access Grants untuk menskalakan izin data S3 untuk pengguna, grup, peran, dan aplikasi.

Untuk sebagian besar kasus penggunaan, Anda dapat mengelola kontrol akses untuk data S3 menggunakan AWS Identity and Access Management (IAM) dengan kebijakan bucket atau kebijakan berbasis identitas IAM.

Namun, jika Anda memiliki persyaratan kontrol akses S3 yang kompleks, seperti berikut ini, Anda bisa mendapatkan keuntungan besar dari menggunakan S3 Access Grants:

- Anda mengalami batas ukuran kebijakan bucket sebesar 20 KB.
- Anda memberikan identitas manusia, misalnya, Microsoft Entra ID (sebelumnya adalah Azure Active Directory), Okta atau pengguna dan grup Ping, akses ke data S3 untuk analitik dan big data.
- Anda harus menyediakan akses lintas akun tanpa sering memperbarui kebijakan IAM.
- Data Anda tidak terstruktur dan tingkat objek daripada terstruktur, dalam format baris dan kolom.

Alur kerja S3 Access Grants adalah sebagai berikut:

Langkah-langkah	Deskripsi
1	Membuat instans S3 Access Grants Untuk memulai, memulai instans S3 Access Grants yang akan berisi pemberian akses individual Anda.
2	Mendaftarkan lokasi Kedua, daftarkan lokasi data S3 (seperti default, `s3://`) dan kemudian tentukan peran IAM default yang diasumsikan oleh S3 Access Grants saat menyediakan akses ke lokasi data S3. Anda juga dapat menambahkan lokasi kustom ke bucket atau prefiks tertentu dan memetakannya ke peran IAM kustom.
3	Membuat pemberian

Langkah-langkah	Deskripsi
	Buat pemberian izin individu. Tentukan dalam izin ini memberikan lokasi S3 terdaftar, ruang lingkup akses data di dalam lokasi, identitas penerima pemberian, dan tingkat aksesnya (`READ`, `WRITE`, atau `READWRITE`).
4	Minta akses ke data S3 Ketika pengguna, aplikasi, dan Layanan AWS ingin mengakses data S3, mereka pertama kali membuat permintaan akses. S3 Access Grants menentukan apakah permintaan harus diotorisasi. Jika ada pemberian terkait yang mengotorisasi akses, S3 Access Grants menggunakan peran IAM lokasi terdaftar yang terkait dengan pemberian tersebut untuk menjual kembali kredensial sementara ke pemohon.
5	Mengakses data S3 Aplikasi menggunakan kredensial sementara yang dijual oleh S3 Access Grants untuk mengakses data S3.

Membuat instans S3 Access Grants

Untuk memulai menggunakan AmazonS3 Access Grants, pertama-tama Anda membuat instans S3 Access Grants. Anda hanya dapat membuat satu instance Hibah Akses S3 Wilayah AWS per akun. Instans S3 Access Grants berfungsi sebagai kontainer untuk sumber daya S3 Access Grants Anda, yang mencakup lokasi dan pemberian terdaftar.

Dengan S3 Access Grants, Anda dapat membuat hibah izin ke data S3 untuk pengguna dan peran AWS Identity and Access Management (IAM). Jika Anda telah menambahkan direktori identitas perusahaan AWS IAM Identity Center, Anda dapat mengaitkan instance IAM Identity Center ini dari direktori perusahaan Anda dengan instans S3 Access Grants Anda. Setelah selesai, Anda dapat membuat pemberian akses untuk pengguna dan grup perusahaan Anda. Jika Anda belum menambahkan direktori perusahaan ke Pusat Identitas IAM, Anda dapat mengaitkan instans S3 Access Grants Anda dengan instans Pusat Identitas IAM nantinya.

Anda dapat membuat instance S3 Access Grants dengan menggunakan konsol Amazon S3,AWS CLI(), AWS Command Line Interface Amazon S3 REST API, dan SDK. AWS

Menggunakan konsol S3

Sebelum Anda dapat memberikan akses ke data S3 Anda dengan S3 Access Grants, Anda harus terlebih dahulu membuat instance S3 Access Grants sama Wilayah AWS dengan data S3 Anda.

Prasyarat

Jika Anda ingin memberikan akses ke data S3 Anda dengan menggunakan identitas dari direktori perusahaan Anda, tambahkan direktori identitas perusahaan Anda ke. AWS IAM Identity Center Jika Anda belum siap untuk melakukannya, Anda dapat mengaitkan instans S3 Access Grants Anda dengan instans IAM Identity Center nanti.

Membuat instans S3 Access Grants

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di bilah navigasi, pilih nama yang saat ini ditampilkan Wilayah AWS. Selanjutnya, pilih Wilayah yang ingin Anda alihkan.
3. Pada panel navigasi di sebelah kiri, pilih Access Grants.
4. Pada halaman S3 Access Grants, pilih Buat S3 Access Grants instans.

   a. Pada Langkah 1 dari wizard Instans Set up Access Grants, verifikasi bahwa Anda ingin membuat instans saat ini Wilayah AWS. Pastikan bahwa ini sama di Wilayah AWS mana data S3 Anda berada. Anda dapat membuat satu instance Hibah Akses S3 Wilayah AWS per akun.

   b. (Opsional) Jika Anda telah menambahkan direktori identitas perusahaan AWS IAM Identity Center, Anda dapat mengaitkan instance IAM Identity Center ini dari direktori perusahaan Anda dengan instans S3 Access Grants Anda.

      Untuk melakukannya, pilih Add IAM Identity Center instans in ***wilayah***. Kemudian masukkan instans Amazon Resource Name (ARN).

      Jika Anda belum menambahkan direktori perusahaan ke Pusat Identitas IAM, Anda dapat mengaitkan instans S3 Access Grants Anda dengan instans Pusat Identitas IAM nantinya.

   c. Untuk membuat instans S3 Access Grants, pilih Berikutnya. Untuk mendaftarkan lokasi, lihat Langkah 2-mendaftarkan lokasi.

5. Jika Selanjutnya atau Buat instans S3 Access Grants dinonaktifkan:

Tidak dapat membuat instans

- Anda mungkin sudah memiliki instans S3 Access Grants Wilayah AWS yang sama. Pada panel navigasi di sebelah kiri, pilih Access Grants. Pada halaman S3 Access Grants, gulir ke bawah ke instans S3 Access Grants di bagian akun Anda untuk menentukan apakah instans sudah ada.
- Anda mungkin tidak memiliki `s3:CreateAccessGrantsInstance` izin yang diperlukan untuk membuat instance S3 Access Grants. Menghubungi administrator akun Anda. Untuk izin tambahan yang diperlukan jika Anda mengaitkan instans Pusat Identitas IAM, dengan instans S3 Access Grants Anda, lihat. CreateAccessGrantsInstance

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Untuk menggunakan perintah contoh berikut, ganti *`user input placeholders`* dengan informasi Anda sendiri.

Example Membuat instans S3 Access Grants

```
aws s3control create-access-grants-instance \
--account-id 111122223333 \
--region us-east-2
```

Respons:

```
{
    "CreatedAt": "2023-05-31T17:54:07.893000+00:00",
    "AccessGrantsInstanceId": "default",
    "AccessGrantsInstanceArn": "arn:aws:s3:us-east-2:111122223333:access-grants/
default"
}
```

Penggunaan API REST

Anda dapat menggunakan API REST Amazon S3 untuk membuat instans S3 Access Grants. Untuk informasi tentang dukungan API REST untuk mengelola instans S3 Access Grants, lihat bagian berikut di Referensi API Amazon Simple Storage Service:

- AssociateAccessGrantsIdentityCenter
- CreateAccessGrantsInstance
- DeleteAccessGrantsInstance
- DissociateAccessGrantsIdentityCenter
- GetAccessGrantsInstance
- GetAccessGrantsInstanceForPrefix
- GetAccessGrantsInstanceResourcePolicy
- ListAccessGrantsInstances
- PutAccessGrantsInstanceResourcePolicy

Menggunakan AWS SDK

Bagian ini memberikan contoh cara membuat instance S3 Access Grants dengan menggunakan SDK AWS .

Java

Contoh ini membuat instans S3 Access Grants, yang berfungsi sebagai kontainer untuk pemberian akses individual Anda. Anda dapat memiliki satu instance S3 Access Grants per akun Wilayah AWS Anda. Respons tersebut mencakup ID instans `default` dan Amazon Resource Name (ARN) yang dihasilkan untuk instans S3 Access Grants Anda.

Example Membuat permintaan instans S3 Access Grants

```
public void createAccessGrantsInstance() {
CreateAccessGrantsInstanceRequest createRequest =
 CreateAccessGrantsInstanceRequest.builder().accountId("111122223333").build();
CreateAccessGrantsInstanceResponse createResponse =
 s3Control.createAccessGrantsInstance(createRequest);LOGGER.info("CreateAccessGrantsInstance
 " + createResponse);
}
```

Respons:

```
CreateAccessGrantsInstanceResponse(
CreatedAt=2023-06-07T01:46:20.507Z,
AccessGrantsInstanceId=default,
AccessGrantsInstanceArn=arn:aws:s3:us-east-2:111122223333:access-grants/default)
```

Topik

- Lihat detail instance S3 Access Grants
- Kaitkan atau pisahkan instans Pusat Identitas IAM Anda
- Hapus instans S3 Access Grants

Lihat detail instance S3 Access Grants

Anda dapat melihat detail instans Hibah Akses Amazon S3 Anda secara khusus. Wilayah AWS Anda juga dapat mencantumkan instans S3 Access Grants, termasuk instance yang telah dibagikan dengan Anda melalui (). AWS Resource Access Manager AWS RAM

Anda dapat melihat detail instans S3 Access Grants atau mencantumkan instans S3 Access Grants dengan menggunakan konsol Amazon S3, (), Amazon S3 REST AWS CLI API AWS Command Line Interface , dan SDK. AWS

Menggunakan konsol S3

Untuk melihat instans S3 Access Grants

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pada panel navigasi di sebelah kiri, pilih Access Grants.
3. Pada halaman S3 Access Grants, pilih Region yang berisi instance S3 Access Grants yang ingin Anda gunakan.
4. Halaman S3 Access Grants mencantumkan instans S3 Access Grants Anda dan instans lintas akun yang telah dibagikan dengan akun Anda. Untuk melihat detail instance, pilih Lihat detail.

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Example — Dapatkan detail instance Hibah Akses S3

```
aws s3control get-access-grants-instance \
```

```
 --account-id 111122223333 \
 --region us-east-2
```

Respons:

```
{
    "AccessGrantsInstanceArn": "arn:aws:s3:us-east-2: 111122223333:access-grants/
default",
    "AccessGrantsInstanceId": "default",
    "CreatedAt": "2023-05-31T17:54:07.893000+00:00"
}
```

Example — Daftar semua instance Hibah Akses S3 untuk akun

Tindakan ini mencantumkan instans S3 Access Grants untuk sebuah akun. Anda hanya dapat memiliki satu instance S3 Access Grants per. Wilayah AWS Tindakan ini juga mencantumkan instance Hibah Akses S3 lintas akun lain yang dapat diakses akun Anda.

```
aws s3control list-access-grants-instances \
 --account-id 111122223333 \
 --region us-east-2
```

Respons:

```
{
    "AccessGrantsInstanceArn": "arn:aws:s3:us-east-2: 111122223333:access-grants/
default",
    "AccessGrantsInstanceId": "default",
    "CreatedAt": "2023-05-31T17:54:07.893000+00:00"
}
```

Penggunaan API REST

Untuk informasi tentang dukungan Amazon S3 REST API untuk mengelola instance S3 Access Grants, lihat bagian berikut di Referensi API Layanan Penyimpanan Sederhana Amazon:

- GetAccessGrantsInstance
- GetAccessGrantsInstanceForPrefix
- ListAccessGrantsInstances

Menggunakan AWS SDK

Bagian ini memberikan contoh cara mendapatkan detail instance S3 Access Grants dengan menggunakan SDK AWS .

Untuk menggunakan contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Java

Example — Dapatkan instance Hibah Akses S3

```
public void getAccessGrantsInstance() {
GetAccessGrantsInstanceRequest getRequest = GetAccessGrantsInstanceRequest.builder()
.accountId("111122223333")
.build();
GetAccessGrantsInstanceResponse getResponse =
 s3Control.getAccessGrantsInstance(getRequest);
LOGGER.info("GetAccessGrantsInstanceResponse: " + getResponse);
}
```

Respons:

```
GetAccessGrantsInstanceResponse(
AccessGrantsInstanceArn=arn:aws:s3:us-east-2: 111122223333:access-grants/default,
CreatedAt=2023-06-07T01:46:20.507Z)
```

Example — Daftar semua instance Hibah Akses S3 untuk akun

Tindakan ini mencantumkan instans S3 Access Grants untuk sebuah akun. Anda hanya dapat memiliki satu instans S3 Access Grants per Wilayah. Tindakan ini juga dapat mencantumkan instans S3 Access Grants lintas akun lain yang dapat diakses oleh akun Anda.

```
public void listAccessGrantsInstances() {
ListAccessGrantsInstancesRequest listRequest =
 ListAccessGrantsInstancesRequest.builder()
.accountId("111122223333")
.build();
ListAccessGrantsInstancesResponse listResponse =
 s3Control.listAccessGrantsInstances(listRequest);
LOGGER.info("ListAccessGrantsInstancesResponse: " + listResponse);
```

```
}
```

Respons:

```
ListAccessGrantsInstancesResponse(
AccessGrantsInstancesList=[
ListAccessGrantsInstanceEntry(
AccessGrantsInstanceId=default,
AccessGrantsInstanceArn=arn:aws:s3:us-east-2:111122223333:access-grants/default,
CreatedAt=2023-06-07T04:28:11.728Z
)
]
)
```

Kaitkan atau pisahkan instans Pusat Identitas IAM Anda

Di Amazon S3 Access Grants, Anda dapat mengaitkan AWS IAM Identity Center instance direktori identitas perusahaan Anda dengan instance S3 Access Grants. Setelah melakukannya, Anda dapat membuat hibah akses untuk pengguna dan grup direktori perusahaan Anda, selain pengguna dan AWS Identity and Access Management peran (IAM).

Jika Anda tidak lagi ingin membuat hibah akses untuk pengguna dan grup direktori perusahaan, Anda dapat memisahkan instans Pusat Identitas IAM Anda dari instans S3 Access Grants Anda.

Anda dapat mengaitkan atau memisahkan instans Pusat Identitas IAM dengan menggunakan konsol Amazon S3,AWS CLI(), AWS Command Line Interface Amazon S3 REST API, dan SDK. AWS

Menggunakan konsol S3

Sebelum Anda mengaitkan instans Pusat Identitas IAM Anda dengan instans S3 Access Grants Anda, Anda harus menambahkan direktori identitas perusahaan Anda ke IAM Identity Center. Untuk informasi selengkapnya, lihat the section called “S3 Access Grants dan identitas direktori perusahaan”.

Untuk mengaitkan instans Pusat Identitas IAM dengan instans S3 Access Grants

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pada panel navigasi di sebelah kiri, pilih Access Grants.

3. Pada halaman S3 Access Grants, pilih Region yang berisi instance S3 Access Grants yang ingin Anda gunakan.
4. Pilih Lihat detail untuk contoh.
5. Pada halaman detail, di bagian Pusat Identitas IAM, pilih untuk Menambahkan instance Pusat Identitas IAM atau Deregister instance Pusat Identitas IAM yang sudah terkait.

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Example — Kaitkan instans Pusat Identitas IAM dengan instance S3 Access Grants

```
aws s3control associate-access-grants-identity-center \
 --account-id 111122223333 \
 --identity-center-arn arn:aws:sso:::instance/ssoins-1234a567bb89012c \
 --profile access-grants-profile \
 --region eu-central-1

// No response body
```

Example — Lepaskan instans Pusat Identitas IAM dari instance S3 Access Grants

```
aws s3control dissociate-access-grants-identity-center \
 --account-id 111122223333 \
 --profile access-grants-profile \
 --region eu-central-1

// No response body
```

Penggunaan API REST

Untuk informasi tentang dukungan Amazon S3 REST API untuk mengelola asosiasi antara instans Pusat Identitas IAM dan instance S3 Access Grants, lihat bagian berikut di Referensi API Layanan Penyimpanan Sederhana Amazon:

- AssociateAccessGrantsIdentityCenter

- DissociateAccessGrantsIdentityCenter

Hapus instans S3 Access Grants

Anda dapat menghapus instans Amazon S3 Access Grants dari akun Wilayah AWS Anda. Namun, sebelum Anda dapat menghapus instance S3 Access Grants, Anda harus terlebih dahulu melakukan hal berikut:

- Hapus semua sumber daya dalam instance S3 Access Grants, termasuk semua hibah dan lokasi. Untuk informasi selengkapnya, lihat Menghapus hibah dan Menghapus lokasi.
- Jika Anda telah mengaitkan AWS IAM Identity Center instance dengan instans S3 Access Grants, Anda harus memisahkan instance IAM Identity Center. Untuk informasi selengkapnya, lihat Mengaitkan atau memisahkan instans Pusat Identitas IAM Anda.

> **Important**
>
> Jika Anda menghapus instance S3 Access Grants, penghapusan bersifat permanen dan tidak dapat dibatalkan. Semua penerima hibah yang diberi akses melalui hibah dalam instance Hibah Akses S3 ini akan kehilangan akses ke data S3 Anda.

Anda dapat menghapus instans S3 Access Grants dengan menggunakan konsol Amazon S3,AWS CLI(), AWS Command Line Interface Amazon S3 REST API, dan SDK. AWS

Menggunakan konsol S3

Untuk menghapus instans S3 Access Grants

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pada panel navigasi di sebelah kiri, pilih Access Grants.
3. Pada halaman S3 Access Grants, pilih Region yang berisi instance S3 Access Grants yang ingin Anda gunakan.
4. Pilih Lihat detail untuk contoh.
5. Pada halaman detail instance, pilih Hapus instance di sudut kanan atas.
6. Di kotak dialog yang muncul, pilih Hapus. Tindakan ini tidak dapat dibatalkan.

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

> **Note**
>
> Sebelum Anda dapat menghapus instance S3 Access Grants, Anda harus terlebih dahulu menghapus semua hibah dan lokasi yang dibuat dalam instance S3 Access Grants. Jika Anda telah mengaitkan instans Pusat Identitas IAM dengan instans S3 Access Grants, Anda harus memutuskannya terlebih dahulu.

Example — Hapus instance Hibah Akses S3

```
aws s3control delete-access-grants-instance \
--account-id 111122223333 \
--profile access-grants-profile \
--region us-east-2 \
--endpoint-url https://s3-control.us-east-2.amazonaws.com \

 // No response body
```

Penggunaan API REST

Untuk informasi tentang dukungan Amazon S3 REST API untuk menghapus instance S3 Access Grants, lihat di Referensi API Layanan DeleteAccessGrantsInstancePenyimpanan Sederhana Amazon.

Menggunakan AWS SDK

Bagian ini memberikan contoh cara menghapus instance S3 Access Grants dengan menggunakan SDK AWS .

Untuk menggunakan contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Java

> **Note**
>
> Sebelum Anda dapat menghapus instance S3 Access Grants, Anda harus terlebih dahulu menghapus semua hibah dan lokasi yang dibuat dalam instance S3 Access Grants. Jika Anda telah mengaitkan instans Pusat Identitas IAM dengan instans S3 Access Grants, Anda harus memutuskannya terlebih dahulu.

Example — Hapus instance Hibah Akses S3

```
public void deleteAccessGrantsInstance() {
DeleteAccessGrantsInstanceRequest deleteRequest =
 DeleteAccessGrantsInstanceRequest.builder()
.accountId("111122223333")
.build();
DeleteAccessGrantsInstanceResponse deleteResponse =
 s3Control.deleteAccessGrantsInstance(deleteRequest);
LOGGER.info("DeleteAccessGrantsInstanceResponse: " + deleteResponse);
}
```

Mendaftarkan lokasi

Setelah membuat instans Amazon S3 Access Grants di akun Anda, Anda dapat mendaftarkan lokasi S3 Wilayah AWS dalam instance tersebut. Lokasi adalah sumber daya S3 yang berisi data yang ingin Anda berikan akses. Anda dapat mendaftarkan lokasi default`s3://`, yang merupakan semua bucket Anda di Wilayah AWS, dan kemudian mempersempit cakupan akses nanti, saat Anda membuat hibah akses individual. Anda juga dapat mendaftarkan bucket tertentu atau bucket dan prefiks sebagai lokasi.

Anda harus terlebih dahulu mendaftarkan setidaknya satu lokasi dengan instans S3 Access Grants Anda sebelum Anda dapat membuat pemberian akses. Saat mendaftarkan lokasi, Anda juga harus menentukan peran AWS Identity and Access Management (IAM) yang diasumsikan oleh S3 Access Grants untuk memenuhi permintaan runtime untuk lokasi dan cakupan izin hingga pemberian tertentu saat runtime.

URI S3	Peran IAM	Deskripsi
`s3://`	*`Default-IAM-role`*	Lokasi default, `s3://`, mencakup semua bucket di file Wilayah AWS.
`s3:`*`//example-s3-bucket1 /`*	*`IAM-role-For-bucket`*	Lokasi ini mencakup semua objek dalam bucket yang ditentukan.

Sebelum Anda dapat mendaftarkan lokasi, pastikan Anda melakukan hal berikut ini:

- Buat satu atau beberapa bucket yang berisi data yang ingin Anda berikan akses. Bucket ini harus ditempatkan Wilayah AWS sama dengan instance S3 Access Grants Anda. Untuk informasi selengkapnya, lihat Membuat bucket.

  Untuk menambahkan prefiks ke bucket, lihat Membuat nama kunci objek.
- Buat peran IAM dan berikan akses pengguna utama layanan S3 Access Grants ke peran ini dalam file kebijakan sumber daya. Untuk melakukannya, Anda dapat membuat file JSON yang berisi pernyataan berikut ini. Untuk menambahkan kebijakan sumber daya ke akun Anda, lihat Membuat dan melampirkan kebijakan yang dikelola pelanggan pertama Anda.

  TestRolePolicy.json

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1234567891011",
      "Action": ["sts:AssumeRole", "sts:SetSourceIdentity", "sts:SetContext"],
      "Effect": "Allow",
      "Principal": {"Service":"access-grants.s3.amazonaws.com"}
    }
  ]
}
```

- Buat kebijakan IAM untuk melampirkan izin Amazon S3 ke peran IAM. Lihat file contoh `iam-policy.json` berikut ini, dan ganti *`user input placeholders`* dengan informasi Anda sendiri.

> **Note**
>
> - Jika Anda menggunakan enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) untuk mengenkripsi data Anda, contoh berikut menyertakan AWS KMS izin yang diperlukan untuk peran IAM dalam kebijakan. Jika Anda tidak menggunakan fitur ini, Anda dapat menghapus izin ini dari kebijakan IAM Anda.
> - Anda dapat membatasi peran IAM untuk mengakses data S3 hanya jika kredensialnya dijual oleh S3 Access Grants. Contoh ini menunjukkan cara menambahkan `Condition` pernyataan untuk instance S3 Access Grants tertentu. Untuk melakukan ini, ganti ARN instance S3 Access Grants dalam pernyataan kondisi dengan ARN instance S3 Access Grants Anda, yang memiliki format: `arn:aws:s3:region:accountId:access-grants/default`

iam-policy.json

```
{
   "Version":"2012-10-17",
   "Statement": [
       {
         "Sid": "ObjectLevelReadPermissions",
         "Effect":"Allow",
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:GetObjectAcl",
            "s3:GetObjectVersionAcl",
            "s3:ListMultipartUploadParts"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition":{
            "StringEquals": { "aws:ResourceAccount": "accountId" },
            "ArnEquals": {
                "s3:AccessGrantsInstanceArn": ["arn:aws:s3:region:accountId:access-
grants/default"]
            }
        }
```

```
      },
      {
         "Sid": "ObjectLevelWritePermissions",
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:PutObjectVersionAcl",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion",
            "s3:AbortMultipartUpload"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition":{
            "StringEquals": { "aws:ResourceAccount": "accountId" },
            "ArnEquals": {
                "s3:AccessGrantsInstanceArn": ["arn:aws:s3:Wilayah
AWS:accountId:access-grants/default"]
            }
         }
      },
      {
         "Sid": "BucketLevelReadPermissions",
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition":{
            "StringEquals": { "aws:ResourceAccount": "accountId" },
            "ArnEquals": {
                "s3:AccessGrantsInstanceArn": ["arn:aws:s3:Wilayah
AWS:accountId:access-grants/default"]
            }
         }
      },
      {
         "Sid": "KMSPermissions",
         "Effect":"Allow",
         "Action":[
```

```
            "kms:Decrypt",
            "kms:GenerateDataKey"
         ],
         "Resource":[
            "*"
         ]
      }
   ]
}
```

Anda dapat mendaftarkan lokasi di instans S3 Access Grants menggunakan konsol Amazon S3,AWS CLI(), AWS Command Line Interface Amazon S3 REST API, atau SDK. AWS

Menggunakan konsol S3

Sebelum Anda dapat memberikan akses ke data S3 Anda dengan S3 Access Grants, Anda harus memiliki setidaknya satu lokasi terdaftar.

Untuk mendaftarkan lokasi di instans S3 Access Grants

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pada panel navigasi di sebelah kiri, pilih Access Grants.
3. Pada halaman S3 Access Grants, pilih Region yang berisi instance S3 Access Grants yang ingin Anda gunakan.

   Jika Anda menggunakan instans S3 Access Grants untuk pertama kalinya, pastikan Anda telah menyelesaikan Langkah 1-buat instans S3 Access Grants dan navigasikan ke Langkah 2 dari wizard instans Set up Access Grants. Jika Anda sudah memiliki instans S3 Access Grants, pilih Lihat detail, lalu dari tab Lokasi, pilih Daftar lokasi.

   a. Untuk lingkup Lokasi, pilih Jelajahi S3 atau masukkan jalur URI S3 ke lokasi yang ingin Anda daftarkan. Untuk format URI S3, lihat tabel format lokasi. Setelah memasukkan URI, Anda dapat memilih Lihat untuk menelusuri lokasi.
   b. Untuk Peran IAM, pilih salah satu hal berikut ini:

      - Pilih dari peran IAM yang ada

        Pilih peran IAM dari daftar dropdown. Setelah Anda memilih peran, pilih Lihat untuk memastikan bahwa peran ini memiliki izin yang diperlukan untuk mengelola lokasi yang

Anda daftarkan. Secara kustom, pastikan bahwa peran ini memberikan S3 Access memberikan izin `sts:AssumeRole` dan `sts:SetSourceIdentity`.

- Masukkan ARN peran IAM

  Arahkan ke Konsol IAM. Salin Nama Sumber Daya Amazon (ARN) dari peran IAM dan tempel di kotak ini.

c. Untuk menyelesaikan, pilih Berikutnya atau Daftar lokasi.

4. Pemecahan Masalah:

   Tidak dapat mendaftarkan lokasi

   - Lokasi mungkin sudah terdaftar.

     Anda mungkin tidak memiliki `s3:CreateAccessGrantsLocation` izin untuk mendaftarkan lokasi. Menghubungi administrator akun Anda.

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Anda dapat mendaftarkan lokasi default, `s3://`, atau lokasi kustom di instans S3 Access Grants Anda. Pastikan Anda terlebih dahulu membuat peran IAM dengan akses pengguna utama ke lokasi, lalu pastikan Anda memberikan izin S3 Access Grants untuk mengambil peran ini.

Untuk menggunakan perintah contoh berikut, ganti *`user input placeholders`* dengan informasi Anda sendiri.

Example Buat kebijakan sumber daya

Buat kebijakan yang memungkinkan S3 Access Grants untuk mengambil peran IAM. Untuk melakukannya, Anda dapat membuat file JSON yang berisi pernyataan berikut ini. Untuk menambahkan kebijakan sumber daya ke akun Anda, lihat Membuat dan melampirkan kebijakan yang dikelola pelanggan pertama Anda.

TestRolePolicy.json

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Sid": "Stmt1234567891011",
      "Action": ["sts:AssumeRole", "sts:SetSourceIdentity"],
      "Effect": "Allow",
      "Principal": {"Service":"access-grants.s3.amazonaws.com"}
    }
  ]
}
```

Example Buat peran

Jalankan perintah IAM berikut untuk membuat peran.

```
aws iam create-role --role-name accessGrantsTestRole \
 --region us-east-2 \
 --assume-role-policy-document file://TestRolePolicy.json
```

Menjalankan `create-role` perintah mengembalikan kebijakan:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "accessGrantsTestRole",
        "RoleId": "AROASRDGX4WM4GH55GIDA",
        "Arn": "arn:aws:iam::111122223333:role/accessGrantsTestRole",
        "CreateDate": "2023-05-31T18:11:06+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "Stmt1685556427189",
                    "Action": [
                        "sts:AssumeRole",
                        "sts:SetSourceIdentity"
                    ],
                    "Effect": "Allow",
                    "Principal": {
                        "Service":"access-grants.s3.amazonaws.com"
                    }
                }
            ]
        }
```

```
    }
}
```

Example

Buat kebijakan IAM untuk melampirkan izin Amazon S3 ke peran IAM. Lihat file contoh `iam-policy.json` berikut ini, dan ganti *`user input placeholders`* dengan informasi Anda sendiri.

> **Note**
>
> Jika Anda menggunakan enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) untuk mengenkripsi data, contoh berikut akan menambahkan AWS KMS izin yang diperlukan untuk peran IAM dalam kebijakan. Jika Anda tidak menggunakan fitur ini, Anda dapat menghapus izin ini dari kebijakan IAM Anda.
> Untuk memastikan bahwa peran IAM hanya dapat digunakan untuk mengakses data di S3 jika kredensialnya dijual oleh S3 Access Grants, contoh ini menunjukkan cara menambahkan pernyataan `Condition` pernyataan yang menentukan instans S3 Access Grants (`s3:AccessGrantsInstance:` *`InstanceArn`*) dalam kebijakan IAM Anda.
> Saat menggunakan contoh kebijakan berikut, ganti *`user input placeholders`* dengan informasi Anda sendiri.

iam-policy.json

```
{
   "Version":"2012-10-17",
   "Statement": [
       {
         "Sid": "ObjectLevelReadPermissions",
         "Effect":"Allow",
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion",
            "s3:GetObjectAcl",
            "s3:GetObjectVersionAcl",
            "s3:ListMultipartUploadParts"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
```

```
         "Condition":{
            "StringEquals": { "aws:ResourceAccount": "accountId" },
            "ArnEquals": {
                "s3:AccessGrantsInstanceArn": ["arn:aws:s3:region:accountId:access-
grants/default"]
            }
        }
      },
      {
         "Sid": "ObjectLevelWritePermissions",
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:PutObjectVersionAcl",
            "s3:DeleteObject",
            "s3:DeleteObjectVersion",
            "s3:AbortMultipartUpload"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition":{
            "StringEquals": { "aws:ResourceAccount": "accountId" },
            "ArnEquals": {
                "s3:AccessGrantsInstanceArn": ["arn:aws:s3:Wilayah
 AWS:accountId:access-grants/default"]
            }
         }
      },
      {
         "Sid": "BucketLevelReadPermissions",
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::*"
         ],
         "Condition":{
            "StringEquals": { "aws:ResourceAccount": "accountId" },
            "ArnEquals": {
                "s3:AccessGrantsInstanceArn": ["arn:aws:s3:Wilayah
 AWS:accountId:access-grants/default"]
```

```
            }
         }
      },
      {
         "Sid": "KMSPermissions",
         "Effect":"Allow",
         "Action":[
            "kms:Decrypt",
            "kms:GenerateDataKey"
         ],
         "Resource":[
            "*"
         ]
      }
   ]
}
```

Example

Jalankan perintah berikut:

```
aws iam put-role-policy \
--role-name accessGrantsTestRole \
--policy-name accessGrantsTestRole \
--policy-document file://iam-policy.json
```

Example Daftarkan lokasi default

```
aws s3control create-access-grants-location \
 --account-id 111122223333 \
 --location-scope s3:// \
 --iam-role-arn arn:aws:iam::111122223333:role/accessGrantsTestRole
```

Respons:

```
{"CreatedAt": "2023-05-31T18:23:48.107000+00:00",
    "AccessGrantsLocationId": "default",
    "AccessGrantsLocationArn": "arn:aws:s3:us-east-2:111122223333:access-grants/
default/location/default",
    "LocationScope": "s3://”
    "IAMRoleArn": "arn:aws:iam::111122223333:role/accessGrantsTestRole"
}
```

Example Daftarkan lokasi kustom

```
aws s3control create-access-grants-location \
 --account-id 111122223333 \
 --location-scope s3://DOC-BUCKET-EXAMPLE/ \
 --iam-role-arn arn:aws:iam::123456789012:role/accessGrantsTestRole
```

Respons:

```
{"CreatedAt": "2023-05-31T18:23:48.107000+00:00",
    "AccessGrantsLocationId": "635f1139-1af2-4e43-8131-a4de006eb456",
    "AccessGrantsLocationArn": "arn:aws:s3:us-east-2: 111122223333:access-grants/
default/location/635f1139-1af2-4e43-8131-a4de006eb888",
    "LocationScope": "s3://DOC-BUCKET-EXAMPLE/",
    "IAMRoleArn": "arn:aws:iam::111122223333:role/accessGrantsTestRole"
}
```

Penggunaan API REST

Untuk informasi tentang dukungan API REST Amazon S3 untuk mengelola instans S3 Access Grants, lihat bagian berikut di Referensi API Amazon Simple Storage Service:

- CreateAccessGrantsLocation
- DeleteAccessGrantsLocation
- GetAccessGrantsLocation
- ListAccessGrantsLocations
- UpdateAccessGrantsLocation

Menggunakan AWS SDK

Bagian ini memberikan contoh cara mendaftarkan lokasi dengan menggunakan AWS SDK.

Untuk menggunakan contoh berikut, ganti *`user input placeholders`* dengan informasi Anda sendiri.

Java

Anda dapat mendaftarkan lokasi default, `s3://`, atau lokasi kustom di instans S3 Access Grants Anda. Pastikan Anda terlebih dahulu membuat peran IAM dengan akses pengguna utama ke lokasi, lalu pastikan Anda memberikan izin S3 Access Grants untuk mengambil peran ini.

Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Example Daftarkan lokasi default

Permintaan:

```
public void createAccessGrantsLocation() {
CreateAccessGrantsLocationRequest createRequest =
 CreateAccessGrantsLocationRequest.builder()
.accountId("111122223333")
.locationScope("s3://")
.iamRoleArn("arn:aws:iam::123456789012:role/accessGrantsTestRole")
.build();
CreateAccessGrantsLocationResponse createResponse =
 s3Control.createAccessGrantsLocation(createRequest);
LOGGER.info("CreateAccessGrantsLocationResponse: " + createResponse);
}
```

Respons:

```
CreateAccessGrantsLocationResponse(
CreatedAt=2023-06-07T04:35:11.027Z,
AccessGrantsLocationId=default,
AccessGrantsLocationArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
location/default,
LocationScope=s3://,
IAMRoleArn=arn:aws:iam::111122223333:role/accessGrantsTestRole
)
```

Example Daftarkan lokasi kustom

Permintaan:

```
public void createAccessGrantsLocation() {
CreateAccessGrantsLocationRequest createRequest =
 CreateAccessGrantsLocationRequest.builder()
.accountId("111122223333")
.locationScope("s3://DOC-BUCKET-EXAMPLE/")
.iamRoleArn("arn:aws:iam::111122223333:role/accessGrantsTestRole")
.build();
CreateAccessGrantsLocationResponse createResponse =
 s3Control.createAccessGrantsLocation(createRequest);
```

```
LOGGER.info("CreateAccessGrantsLocationResponse: " + createResponse);
}
```

Respons:

```
CreateAccessGrantsLocationResponse(
CreatedAt=2023-06-07T04:35:10.027Z,
AccessGrantsLocationId=18cfe6fb-eb5a-4ac5-aba9-8d79f04c2012,
AccessGrantsLocationArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
location/18cfe6fb-eb5a-4ac5-aba9-8d79f04c2666,
LocationScope= s3://test-bucket-access-grants-user123/,
IAMRoleArn=arn:aws:iam::111122223333:role/accessGrantsTestRole
)
```

Topik

- Lihat detail lokasi yang terdaftar
- Memperbarui lokasi terdaftar
- Hapus lokasi terdaftar

Lihat detail lokasi yang terdaftar

Anda bisa mendapatkan detail lokasi yang terdaftar di instans S3 Access Grants dengan menggunakan konsol Amazon S3,AWS CLI(), AWS Command Line Interface Amazon S3 REST API, dan SDK. AWS

Menggunakan konsol S3

Untuk melihat lokasi yang terdaftar di instans S3 Access Grants

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pada panel navigasi di sebelah kiri, pilih Access Grants.
3. Pada halaman S3 Access Grants, pilih Region yang berisi instance S3 Access Grants yang ingin Anda gunakan.
4. Pilih Lihat detail untuk contoh.
5. Pada halaman detail untuk contoh, pilih tab Lokasi.

6. Temukan lokasi terdaftar yang ingin Anda lihat. Untuk memfilter daftar lokasi terdaftar, gunakan kotak pencarian.

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Example — Dapatkan detail lokasi yang terdaftar

```
aws s3control get-access-grants-location \
--account-id 111122223333 \
--access-grants-location-id default
```

Respons:

```
{
    "CreatedAt": "2023-05-31T18:23:48.107000+00:00",
    "AccessGrantsLocationId": "default",
    "AccessGrantsLocationArn": "arn:aws:s3:us-east-2:111122223333:access-grants/
default/location/default",
    "IAMRoleArn": "arn:aws:iam::111122223333:role/accessGrantsTestRole"
}
```

Example — Daftar semua lokasi yang terdaftar dalam instance S3 Access Grants

Untuk membatasi hasil ke awalan atau bucket S3, Anda dapat menggunakan parameter secara opsional. `--location-scope s3://`*bucket-and-or-prefix*

```
aws s3control list-access-grants-locations \
--account-id 111122223333 \
--region us-east-2
```

Respons:

```
{"AccessGrantsLocationsList": [
  {
    "CreatedAt": "2023-05-31T18:23:48.107000+00:00",
```

```
    "AccessGrantsLocationId": "default",
    "AccessGrantsLocationArn": "arn:aws:s3:us-east-2:111122223333:access-grants/
default/location/default",
    "LocationScope": "s3://"
    "IAMRoleArn": "arn:aws:iam::111122223333:role/accessGrantsTestRole"
     },
  {
    "CreatedAt": "2023-05-31T18:23:48.107000+00:00",
    "AccessGrantsLocationId": "635f1139-1af2-4e43-8131-a4de006eb456",
    "AccessGrantsLocationArn": "arn:aws:s3:us-east-2:111122223333:access-grants/
default/location/635f1139-1af2-4e43-8131-a4de006eb888",
    "LocationScope": "s3://DOC-EXAMPLE-BUCKET/prefixA*",
    "IAMRoleArn": "arn:aws:iam::111122223333:role/accessGrantsTestRole"
     }
   ]
  }
```

Penggunaan API REST

Untuk informasi tentang dukungan Amazon S3 REST API untuk mendapatkan detail lokasi terdaftar atau mencantumkan semua lokasi yang terdaftar dengan instans S3 Access Grants, lihat bagian berikut di Referensi API Layanan Penyimpanan Sederhana Amazon:

- GetAccessGrantsLocation
- ListAccessGrantsLocations

Menggunakan AWS SDK

Bagian ini memberikan contoh cara mendapatkan detail lokasi terdaftar atau mencantumkan semua lokasi terdaftar di instance S3 Access Grants dengan menggunakan SDK. AWS

Untuk menggunakan contoh berikut, ganti `user input placeholders` dengan informasi Anda sendiri.

Java

Example — Dapatkan detail lokasi yang terdaftar

```
public void getAccessGrantsLocation() {
GetAccessGrantsLocationRequest getAccessGrantsLocationRequest =
 GetAccessGrantsLocationRequest.builder()
.accountId("111122223333")
```

```
.accessGrantsLocationId("default")
.build();
GetAccessGrantsLocationResponse getAccessGrantsLocationResponse =
 s3Control.getAccessGrantsLocation(getAccessGrantsLocationRequest);
LOGGER.info("GetAccessGrantsLocationResponse: " + getAccessGrantsLocationResponse);
}
```

Respons:

```
GetAccessGrantsLocationResponse(
CreatedAt=2023-06-07T04:35:10.027Z,
AccessGrantsLocationId=default,
AccessGrantsLocationArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
location/default,
LocationScope= s3://,
IAMRoleArn=arn:aws:iam::111122223333:role/accessGrantsTestRole
)
```

Example — Daftar semua lokasi terdaftar dalam instance S3 Access Grants

Untuk membatasi hasil ke awalan atau bucket S3, Anda dapat meneruskan URI S3 secara opsional, sepertis`3://`*`bucket-and-or-prefix`*, dalam parameter. `LocationScope`

```
public void listAccessGrantsLocations() {

ListAccessGrantsLocationsRequest listRequest =
 ListAccessGrantsLocationsRequest.builder()
.accountId("111122223333")
.build();

ListAccessGrantsLocationsResponse listResponse =
 s3Control.listAccessGrantsLocations(listRequest);
LOGGER.info("ListAccessGrantsLocationsResponse: " + listResponse);
}
```

Respons:

```
ListAccessGrantsLocationsResponse(
AccessGrantsLocationsList=[
ListAccessGrantsLocationsEntry(
CreatedAt=2023-06-07T04:35:11.027Z,
AccessGrantsLocationId=default,
```

```
AccessGrantsLocationArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
location/default,
LocationScope=s3://,
IAMRoleArn=arn:aws:iam::111122223333:role/accessGrantsTestRole
),
ListAccessGrantsLocationsEntry(
CreatedAt=2023-06-07T04:35:10.027Z,
AccessGrantsLocationId=635f1139-1af2-4e43-8131-a4de006eb456,
AccessGrantsLocationArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
location/635f1139-1af2-4e43-8131-a4de006eb888,
LocationScope=s3://DOC-EXAMPLE-BUCKET/prefixA*,
IAMRoleArn=arn:aws:iam::111122223333:role/accessGrantsTestRole
)
]
)
```

Memperbarui lokasi terdaftar

Anda dapat memperbarui peran AWS Identity and Access Management (IAM) lokasi yang terdaftar di instans Amazon S3 Access Grants. Untuk setiap peran IAM baru yang Anda gunakan untuk mendaftarkan lokasi di S3 Access Grants, pastikan untuk memberikan akses utama layanan S3 Access Grants (`access-grants.s3.amazonaws.com`) ke peran ini. Untuk melakukannya, tambahkan entri untuk peran IAM baru dalam file JSON kebijakan kepercayaan yang sama yang Anda gunakan saat pertama kali mendaftarkan lokasi.

Anda dapat memperbarui lokasi di instans S3 Access Grants dengan menggunakan konsol Amazon S3,AWS CLI(), AWS Command Line Interface Amazon S3 REST API, dan SDK. AWS

Menggunakan konsol S3

Untuk memperbarui peran IAM dari lokasi yang terdaftar dengan instans S3 Access Grants

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pada panel navigasi di sebelah kiri, pilih Access Grants.
3. Pada halaman S3 Access Grants, pilih Region yang berisi instance S3 Access Grants yang ingin Anda gunakan.
4. Pilih Lihat detail untuk contoh.
5. Pada halaman detail untuk contoh, pilih tab Lokasi.

6. Temukan lokasi yang ingin Anda perbarui. Untuk memfilter daftar lokasi, gunakan kotak pencarian.
7. Pilih tombol opsi di sebelah lokasi terdaftar yang ingin Anda perbarui.
8. Perbarui peran IAM, lalu pilih Simpan perubahan.

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Example — Perbarui peran IAM dari lokasi terdaftar

```
aws s3control update-access-grants-location \
--account-id 111122223333 \
--access-grants-location-id 635f1139-1af2-4e43-8131-a4de006eb999 \
--iam-role-arn arn:aws:iam::777788889999:role/accessGrantsTestRole
```

Respons:

```
{
    "CreatedAt": "2023-05-31T18:23:48.107000+00:00",
    "AccessGrantsLocationId": "635f1139-1af2-4e43-8131-a4de006eb999",
    "AccessGrantsLocationArn": "arn:aws:s3:us-east-2:777788889999:access-grants/
default/location/635f1139-1af2-4e43-8131-a4de006eb888",
    "LocationScope": "s3://DOC-EXAMPLE-BUCKET/prefixB*",
    "IAMRoleArn": "arn:aws:iam::777788889999:role/accessGrantsTestRole"
}
```

Penggunaan API REST

Untuk informasi tentang dukungan Amazon S3 REST API untuk memperbarui lokasi di instans S3 Access Grants, lihat di Referensi API UpdateAccessGrantsLocationLayanan Penyimpanan Sederhana Amazon.

Menggunakan AWS SDK

Bagian ini memberikan contoh cara memperbarui peran IAM dari lokasi terdaftar dengan menggunakan AWS SDK.

Untuk menggunakan contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Java

Example — Perbarui peran IAM dari lokasi terdaftar

```
public void updateAccessGrantsLocation() {
UpdateAccessGrantsLocationRequest updateRequest =
 UpdateAccessGrantsLocationRequest.builder()
.accountId("111122223333")
.accessGrantsLocationId("635f1139-1af2-4e43-8131-a4de006eb999")
.iamRoleArn("arn:aws:iam::777788889999:role/accessGrantsTestRole")
.build();
UpdateAccessGrantsLocationResponse updateResponse =
 s3Control.updateAccessGrantsLocation(updateRequest);
LOGGER.info("UpdateAccessGrantsLocationResponse: " + updateResponse);
}
```

Respons:

```
UpdateAccessGrantsLocationResponse(
CreatedAt=2023-06-07T04:35:10.027Z,
AccessGrantsLocationId=635f1139-1af2-4e43-8131-a4de006eb999,
AccessGrantsLocationArn=arn:aws:s3:us-east-2:777788889999:access-grants/default/
location/635f1139-1af2-4e43-8131-a4de006eb888,
LocationScope=s3://DOC-EXAMPLE-BUCKET/prefixB*,
IAMRoleArn=arn:aws:iam::777788889999:role/accessGrantsTestRole
)
```

Hapus lokasi terdaftar

Anda dapat menghapus pendaftaran lokasi dari instans Amazon S3 Access Grants. Menghapus lokasi membatalkan pendaftarannya dari instance S3 Access Grants.

Sebelum Anda dapat menghapus pendaftaran lokasi dari instance S3 Access Grants, Anda harus menghapus semua hibah yang terkait dengan lokasi ini. Untuk selengkapnya tentang cara menghapus hibah, lihat Menghapus hibah.

Anda dapat menghapus lokasi di instans S3 Access Grants dengan menggunakan konsol Amazon S3,AWS CLI(), AWS Command Line Interface Amazon S3 REST API, dan SDK. AWS

Menggunakan konsol S3

Untuk menghapus pendaftaran lokasi dari instans S3 Access Grants

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pada panel navigasi di sebelah kiri, pilih Access Grants.
3. Pada halaman S3 Access Grants, pilih Region yang berisi instance S3 Access Grants yang ingin Anda gunakan.
4. Pilih Lihat detail untuk contoh.
5. Pada halaman detail untuk contoh, pilih tab Lokasi.
6. Temukan lokasi yang ingin Anda perbarui. Untuk memfilter daftar lokasi, gunakan kotak pencarian.
7. Pilih tombol opsi di sebelah lokasi terdaftar yang ingin Anda hapus.
8. Pilih Batalkan pendaftaran.
9. Kotak dialog muncul yang memperingatkan Anda bahwa tindakan ini tidak dapat dibatalkan. Untuk menghapus lokasi, pilih Deregister.

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Example — Hapus pendaftaran lokasi

```
aws s3control delete-access-grants-location \
--account-id 111122223333 \
--access-grants-location-id  a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
 // No response body
```

Penggunaan API REST

Untuk informasi tentang dukungan Amazon S3 REST API untuk menghapus lokasi dari instance S3 Access Grants, lihat di DeleteAccessGrantsLocationReferensi API Layanan Penyimpanan Sederhana Amazon.

Menggunakan AWS SDK

Bagian ini memberikan contoh cara menghapus lokasi dengan menggunakan AWS SDK.

Untuk menggunakan contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Java

Example — Hapus pendaftaran lokasi

```
public void deleteAccessGrantsLocation() {
DeleteAccessGrantsLocationRequest deleteRequest =
 DeleteAccessGrantsLocationRequest.builder()
.accountId("111122223333")
.accessGrantsLocationId("a1b2c3d4-5678-90ab-cdef-EXAMPLE11111")
.build();
DeleteAccessGrantsLocationResponse deleteResponse =
 s3Control.deleteAccessGrantsLocation(deleteRequest);
LOGGER.info("DeleteAccessGrantsLocationResponse: " + deleteResponse);
}
```

Respons:

```
DeleteAccessGrantsLocationResponse()
```

Membuat pemberian

Setelah mendaftarkan setidaknya satu lokasi di instans Amazon S3 Access Grants, Anda dapat membuat pemberian akses. Hibah akses memberikan izin penerima pemberian untuk mengakses lokasi terdaftar.

Penerima hibah dapat berupa pengguna AWS Identity and Access Management (IAM) atau peran atau pengguna direktori atau grup. Pengguna direktori adalah pengguna dari direktori perusahaan atau sumber identitas eksternal yang Anda tambahkan ke dalam AWS IAM Identity Center instans yang terkait dengan instans S3 Access Grants Anda. Untuk membuat pemberian untuk pengguna atau grup tertentu dari IAM Identity Center, temukan GUID yang digunakan IAM Identity Center untuk mengidentifikasi pengguna tersebut di Pusat Identitas IAM, misalnya, `a1b2c3d4-5678-90ab-cdef-EXAMPLE11111`.

Anda dapat memberikan akses ke sebuah bucket, prefiks, atau objek. Prefiks di Amazon S3 adalah string karakter di awal nama kunci objek yang digunakan untuk mengatur objek dalam bucket. Ini dapat berupa string karakter yang diizinkan, misalnya, nama kunci objek dalam bucket yang dimulai dengan prefiks `engineering/`.

Subprefiks

Saat memberikan akses ke lokasi terdaftar, Anda dapat menggunakan bidang `Subprefix` untuk mempersempit cakupan ke prefiks tertentu dalam bucket atau objek tertentu dalam bucket.

Anda tidak dapat membuat pemberian akses untuk lokasi `s3://` default, yang akan memberikan akses kepada penerima pemberian ke setiap bucket di dalam Wilayah. Jika Anda memilih lokasi default `s3://` sebagai lokasi pemberian, Anda harus mempersempit cakupan pemberian dengan menggunakan bidang `Subprefix` untuk menentukan salah satu dari berikut ini:

- Sebuah bucket—`s3://`*`bucket`*`/*`
- Awalan dalam bucket—`s3://`*`bucket`*`/`*`prefix`*`*`
- Sebuah prefiks di dalam prefiks—`s3://`*`bucket`*`/`*`prefixA`*`/`*`prefixB`*`*`
- Sebuah objek—`s3://`*`bucket`*`/`*`object-key-name`*

Jika Anda membuat pemberian akses di mana lokasi terdaftar adalah bucket, Anda dapat meneruskan salah satu dari yang berikut ini di `Subprefix` bidang:

- Awalan di dalam bucket—*`prefix`*`*`
- Sebuah prefiks di dalam prefiks—*`prefixA/prefixB`*`*`
- Sebuah objek—`/`*`object-key-name`*

Cakupan hibah yang ditampilkan di konsol Amazon S3 atau `GrantScope` yang dikembalikan dalam respons API atau AWS Command Line Interface (AWS CLI) adalah hasil dari penggabungan jalur lokasi dengan. `Subprefix` Pastikan jalur gabungan ini dipetakan dengan benar ke bucket, prefiks, atau S3 Object yang ingin Anda berikan aksesnya.

Jika Anda membuat pemberian akses yang memberikan akses hanya ke satu objek, tentukan dalam panggilan API atau perintah CLI yang menunjukkan bahwa `s3PrefixType` adalah `Object`.

> **Note**
>
> Anda tidak dapat membuat pemberian ke bucket jika bucket belum ada. Namun, Anda dapat membuat pemberian ke prefiks yang belum ada.

Anda dapat membuat hibah akses dengan menggunakan konsol Amazon S3, API Amazon S3 REST AWS CLI, dan SDK. AWS

Menggunakan konsol S3

Untuk membuat pemberian akses

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pada panel navigasi di sebelah kiri, pilih Access Grants.
3. Pada halaman S3 Access Grants, pilih Region yang berisi instance S3 Access Grants yang ingin Anda gunakan.

   Jika Anda menggunakan instans S3 Access Grants untuk pertama kalinya, pastikan Anda telah menyelesaikan Langkah 2-daftarkan lokasi dan navigasikan ke Langkah 3 dari wizard Buat instans Access Grants. Jika Anda sudah memiliki instans S3 Access Grants, pilih Lihat detail, lalu dari tab Izin, pilih Buat pemberian.

   a. Di bagian Cakupan pemberian, pilih atau masukkan lokasi yang telah terdaftar.

      Jika Anda memilih lokasi default `s3://`, gunakan kotak Subprefiks untuk mempersempit cakupan pemberian akses. Untuk informasi selengkapnya, lihat Subprefiks. Jika Anda memberikan akses hanya ke objek, pilih Ruang lingkup pemberian adalah sebuah objek.

   b. Di bawah Izin dan akses, pilih tingkat Izin, baik Baca, Tulis, atau keduanya.

      Kemudian pilih jenis Jenis pemberian. Jika Anda telah menambahkan direktori perusahaan Anda ke IAM Identity Center dan mengaitkan instans IAM Identity Center ini dengan instans S3 Access Grants Anda, Anda dapat memilih Identitas direktori dari IAM Identity Center. Jika Anda memilih opsi ini, dapatkan ID pengguna atau grup dari IAM Identity Center dan masukkan di bagian ini.

Jika Jenis penerima adalah pengguna IAM atau peran, pilih Pengguna utama IAM. Di bawah Jenis pengguna utama IAM, pilih Pengguna atau Peran. Kemudian, di bawah Pengguna utama IAM, pilih dari daftar atau masukkan ID identitas.

c. Untuk membuat pemberian S3 Access Grants, pilih Selanjutnya atau Buat pemberian.

4. Jika Selanjutnya atau Buat pemberian dinonaktifkan:

   Tidak dapat membuat pemberian

   - Anda mungkin perlu mendaftarkan lokasi terlebih dahulu di dalam instans S3 Access Grants Anda.
   - Anda mungkin tidak memiliki izin `s3:CreateAccessGrant` untuk membuat akses. Menghubungi administrator akun Anda.

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Contoh berikut menunjukkan cara membuat permintaan pemberian akses untuk pengguna utama IAM dan cara membuat permintaan pemberian akses untuk pengguna atau grup direktori perusahaan.

Untuk menggunakan perintah contoh berikut, ganti *`user input placeholders`* dengan informasi Anda sendiri.

> **Note**
>
> Jika Anda membuat pemberian akses yang memberikan akses hanya ke satu objek, sertakan parameter `--s3-prefix-type Object` yang diperlukan.

Example Buat permintaan pemberian akses untuk pengguna utama IAM

```
aws s3control create-access-grant \
--account-id 111122223333 \
--access-grants-location-id a1b2c3d4-5678-90ab-cdef-EXAMPLE22222 \
--access-grants-location-configuration S3SubPrefix=prefixB* \
--permission READ \
```

```
--grantee GranteeType=IAM,GranteeIdentifier=arn:aws:iam::123456789012:user/data-consumer-3
```

Example Membuat respons pemberian akses

```
{"CreatedAt": "2023-05-31T18:41:34.663000+00:00",
    "AccessGrantId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "AccessGrantArn": "arn:aws:s3:us-east-2:111122223333:access-grants/default/grant/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "Grantee": {
        "GranteeType": "IAM",
        "GranteeIdentifier": "arn:aws:iam::111122223333:user/data-consumer-3"
    },
    "AccessGrantsLocationId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "AccessGrantsLocationConfiguration": {
        "S3SubPrefix": "prefixB*"
    },
    "GrantScope": "s3://DOC-BUCKET-EXAMPLE/prefix*",
    "Permission": "READ"
}
```

Membuat permintaan pemberian akses untuk pengguna atau grup direktori

Untuk membuat permintaan pemberian akses untuk pengguna atau grup direktori, Anda harus terlebih dahulu mendapatkan GUID untuk pengguna direktori atau grup dengan menjalankan salah satu perintah berikut.

Example Dapatkan GUID untuk pengguna direktori atau grup

Anda dapat menemukan GUID pengguna IAM Identity Center melalui konsol IAM Identity Center atau dengan menggunakan atau SDK AWS CLI . AWS Perintah berikut mencantumkan pengguna dalam contoh instans IAM Identity Center yang ditentukan, dengan nama dan pengenal mereka.

```
aws identitystore list-users --identity-store-id d-1a2b3c4d1234
```

Perintah ini mencantumkan grup dalam instans IAM Identity Center yang ditentukan.

```
aws identitystore list-groups --identity-store-id d-1a2b3c4d1234
```

Example Membuat pemberian akses untuk pengguna atau grup direktori

Perintah ini mirip dengan membuat pemberian untuk pengguna atau peran IAM, kecuali jenis penerima pemberian adalah `DIRECTORY_USER` atau `DIRECTORY_GROUP`, dan pengidentifikasi penerima pemberian adalah GUID untuk pengguna direktori atau grup.

```
aws s3control create-access-grant \
--account-id 123456789012 \
--access-grants-location-id default \
--access-grants-location-configuration S3SubPrefix="DOC-EXAMPLE-BUCKET/rafael/*" \
--permission READWRITE \
--grantee GranteeType=DIRECTORY_USER,GranteeIdentifier=83d43802-00b1-7054-db02-
f1d683aacba5 \
```

Penggunaan API REST

Untuk informasi tentang support API REST Amazon S3 untuk mengelola pemberian akses, lihat bagian berikut di Referensi API Amazon Storage Service:

- CreateAccessGrant
- DeleteAccessGrant
- GetAccessGrant
- ListAccessGrants

Menggunakan AWS SDK

Bagian ini memberikan contoh cara membuat pemberian akses dengan menggunakan SDK AWS .

Java

Untuk menggunakan contoh berikut, ganti *`user input placeholders`* dengan informasi Anda sendiri:

> **Note**
>
> Jika Anda membuat pemberian akses yang memberikan akses hanya ke satu objek, sertakan parameter `.s3PrefixType(S3PrefixType.Object)` yang diperlukan.

Example Buat permintaan pemberian akses

```
public void createAccessGrant() {
CreateAccessGrantRequest createRequest = CreateAccessGrantRequest.builder()
.accountId("111122223333")
.accessGrantsLocationId("a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa")
.permission("READ")
.accessGrantsLocationConfiguration(AccessGrantsLocationConfiguration.builder().s3SubPrefix("
.grantee(Grantee.builder().granteeType("IAM").granteeIdentifier("arn:aws:iam::111122223333:u
data-consumer-3").build())
.build();
CreateAccessGrantResponse createResponse =
 s3Control.createAccessGrant(createRequest);
LOGGER.info("CreateAccessGrantResponse: " + createResponse);
}
```

Example Membuat respons pemberian akses

```
CreateAccessGrantResponse(
CreatedAt=2023-06-07T05:20:26.330Z,
AccessGrantId=a1b2c3d4-5678-90ab-cdef-EXAMPLE33333,
AccessGrantArn=arn:aws:s3:us-east-2:444455556666:access-grants/default/grant/
a1b2c3d4-5678-90ab-cdef-EXAMPLE33333,
Grantee=Grantee(
GranteeType=IAM,
GranteeIdentifier=arn:aws:iam::111122223333:user/data-consumer-3
),
AccessGrantsLocationId=a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa,
AccessGrantsLocationConfiguration=AccessGrantsLocationConfiguration(
S3SubPrefix=prefixB*
),
GrantScope=s3://DOC-BUCKET-EXAMPLE/prefixB,
Permission=READ
)
```

Topik

- Lihat izin
- Hapus izin

Lihat izin

Anda dapat melihat detail pemberian akses di instans Amazon S3 Access Grants dengan menggunakan konsol Amazon S3, AWS Command Line Interface (), Amazon S3 AWS CLI REST API, dan SDK. AWS

Menggunakan konsol S3

Untuk melihat detail hibah akses

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pada panel navigasi di sebelah kiri, pilih Access Grants.
3. Pada halaman S3 Access Grants, pilih Region yang berisi instance S3 Access Grants yang ingin Anda gunakan.
4. Pilih Lihat detail untuk contoh.
5. Pada halaman detail, pilih tab Hibah.
6. Di bagian Hibah, temukan akses hibah yang ingin Anda lihat. Untuk memfilter daftar hibah, gunakan kotak pencarian.

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Example — Dapatkan detail hibah akses

```
aws s3control get-access-grant \
--account-id 111122223333 \
--access-grant-id a1b2c3d4-5678-90ab-cdef-EXAMPLE22222
```

Respons:

```
{
    "CreatedAt": "2023-05-31T18:41:34.663000+00:00",
```

```
    "AccessGrantId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "AccessGrantArn": "arn:aws:s3:us-east-2:111122223333:access-grants/default/
grant-a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "Grantee": {
        "GranteeType": "IAM",
        "GranteeIdentifier": "arn:aws:iam::111122223333:user/data-consumer-3"
    },
    "Permission": "READ",
    "AccessGrantsLocationId": "12a6710f-5af8-41f5-b035-0bc795bf1a2b",
    "AccessGrantsLocationConfiguration": {
        "S3SubPrefix": "prefixB*"
    },
    "GrantScope": "s3://DOC-EXAMPLE-BUCKET/"
}
```

Example — Daftar semua hibah akses dalam instance S3 Access Grants

Anda dapat menggunakan parameter berikut secara opsional untuk membatasi hasil ke awalan S3 atau identitas AWS Identity and Access Management (IAM):

- Subprefiks - `--grant-scope s3://bucket-name/prefix*`
- Identitas IAM — `--grantee-type IAM` dan `--grantee-identifier arn:aws:iam::123456789000:role/accessGrantsConsumerRole`

```
aws s3control list-access-grants \
--account-id 111122223333
```

Respons:

```
{
    "AccessGrantsList": [{"CreatedAt": "2023-06-14T17:54:46.542000+00:00",
            "AccessGrantId": "dd8dd089-b224-4d82-95f6-975b4185bbaa",
            "AccessGrantArn": "arn:aws:s3:us-east-2:111122223333:access-grants/default/
grant/dd8dd089-b224-4d82-95f6-975b4185bbaa",
            "Grantee": {
                "GranteeType": "IAM",
                "GranteeIdentifier": "arn:aws:iam::111122223333:user/data-consumer-3"
            },
            "Permission": "READ",
            "AccessGrantsLocationId": "23514a34-ea2e-4ddf-b425-d0d4bfcarda1",
            "GrantScope": "s3://DOC-EXAMPLE-BUCKET/prefixA*"
```

```
        },
        {"CreatedAt": "2023-06-24T17:54:46.542000+00:00",
            "AccessGrantId": "ee8ee089-b224-4d72-85f6-975b4185a1b2",
            "AccessGrantArn": "arn:aws:s3:us-east-2:111122223333:access-grants/default/
grant/ee8ee089-b224-4d72-85f6-975b4185a1b2",
            "Grantee": {
                "GranteeType": "IAM",
                "GranteeIdentifier": "arn:aws:iam::111122223333:user/data-consumer-9"
            },
            "Permission": "READ",
            "AccessGrantsLocationId": "12414a34-ea2e-4ddf-b425-d0d4bfcacao0",
            "GrantScope": "s3://DOC-EXAMPLE-BUCKET/prefixB*"
        },

    ]
}
```

Penggunaan API REST

Anda dapat menggunakan operasi Amazon S3 API untuk melihat detail pemberian akses dan mencantumkan semua hibah akses dalam instance S3 Access Grants. Untuk informasi tentang dukungan REST API untuk mengelola hibah akses, lihat bagian berikut di Referensi API Amazon Simple Storage Service:

- GetAccessGrant
- ListAccessGrants

Menggunakan AWS SDK

Bagian ini memberikan contoh cara mendapatkan rincian hibah akses dengan menggunakan AWS SDK.

Untuk menggunakan contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Java

Example — Dapatkan detail hibah akses

```
public void getAccessGrant() {
```

```
GetAccessGrantRequest getRequest = GetAccessGrantRequest.builder()
.accountId("111122223333")
.accessGrantId("a1b2c3d4-5678-90ab-cdef-EXAMPLE22222")
.build();
GetAccessGrantResponse getResponse = s3Control.getAccessGrant(getRequest);
LOGGER.info("GetAccessGrantResponse: " + getResponse);
}
```

Respons:

```
GetAccessGrantResponse(
CreatedAt=2023-06-07T05:20:26.330Z,
AccessGrantId=a1b2c3d4-5678-90ab-cdef-EXAMPLE22222,
AccessGrantArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
grant-fd3a5086-42f7-4b34-9fad-472e2942c70e,
Grantee=Grantee(
GranteeType=IAM,
GranteeIdentifier=arn:aws:iam::111122223333:user/data-consumer-3
),
Permission=READ,
AccessGrantsLocationId=12a6710f-5af8-41f5-b035-0bc795bf1a2b,
AccessGrantsLocationConfiguration=AccessGrantsLocationConfiguration(
S3SubPrefix=prefixB*
),
GrantScope=s3://DOC-EXAMPLE-BUCKET/
)
```

Example — Daftar semua hibah akses dalam instance S3 Access Grants

Anda dapat menggunakan parameter ini secara opsional untuk membatasi hasil ke prefiks S3, atau identitas IAM:

- Lingkup - `GrantScope=s3://bucket-name/prefix*`
- Penerima hibah — dan `GranteeType=IAM` `GranteeIdentifier=arn:aws:iam::111122223333:role/accessGrantsConsumerRole`

```
public void listAccessGrants() {
ListAccessGrantsRequest listRequest = ListAccessGrantsRequest.builder()
.accountId("111122223333")
.build();
```

```
ListAccessGrantsResponse listResponse = s3Control.listAccessGrants(listRequest);
LOGGER.info("ListAccessGrantsResponse: " + listResponse);
}
```

Respons:

```
ListAccessGrantsResponse(
AccessGrantsList=[
ListAccessGrantEntry(
CreatedAt=2023-06-14T17:54:46.540z,
AccessGrantId=dd8dd089-b224-4d82-95f6-975b4185bbaa,
AccessGrantArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
grant/dd8dd089-b224-4d82-95f6-975b4185bbaa,
Grantee=Grantee(
GranteeType=IAM, GranteeIdentifier= arn:aws:iam::111122223333:user/data-consumer-3
),
Permission=READ,
AccessGrantsLocationId=23514a34-ea2e-4ddf-b425-d0d4bfcarda1,
GrantScope=s3://DOC-EXAMPLE-BUCKET/prefixA
),
ListAccessGrantEntry(
CreatedAt=2023-06-24T17:54:46.540Z,
AccessGrantId=ee8ee089-b224-4d72-85f6-975b4185a1b2,
AccessGrantArn=arn:aws:s3:us-east-2:111122223333:access-grants/default/
grant/ee8ee089-b224-4d72-85f6-975b4185a1b2,
Grantee=Grantee(
GranteeType=IAM, GranteeIdentifier= arn:aws:iam::111122223333:user/data-consumer-9
),
Permission=READ,
AccessGrantsLocationId=12414a34-ea2e-4ddf-b425-d0d4bfcacao0,
GrantScope=s3://DOC-EXAMPLE-BUCKET/prefixB*
)
]
)
```

Hapus izin

Anda dapat menghapus hibah akses dari instans Amazon S3 Access Grants. Anda tidak dapat membatalkan penghapusan hibah akses. Setelah Anda menghapus hibah akses, penerima hibah tidak akan lagi memiliki akses ke data Amazon S3 Anda.

Anda dapat menghapus hibah akses dengan menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), Amazon S3 REST API, dan SDK. AWS

Menggunakan konsol S3

Untuk menghapus hibah akses

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pada panel navigasi di sebelah kiri, pilih Access Grants.
3. Pada halaman S3 Access Grants, pilih Region yang berisi instance S3 Access Grants yang ingin Anda gunakan.
4. Pilih Lihat detail untuk contoh.
5. Pada halaman detail, pilih tab Hibah.
6. Cari hibah yang ingin Anda hapus. Saat Anda menemukan hibah, pilih tombol radio di sebelahnya.
7. Pilih Hapus. Kotak dialog muncul dengan peringatan bahwa tindakan Anda tidak dapat dibatalkan. Pilih Hapus lagi untuk menghapus hibah.

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Example — Hapus hibah akses

```
aws s3control delete-access-grant \
--account-id 111122223333 \
--access-grant-id a1b2c3d4-5678-90ab-cdef-EXAMPLE11111

// No response body
```

Penggunaan API REST

Untuk informasi tentang dukungan Amazon S3 REST API untuk mengelola hibah akses, lihat DeleteAccessGrantdi Referensi API Amazon Simple Storage Service.

Menggunakan AWS SDK

Bagian ini memberikan contoh cara menghapus akses hibah dengan menggunakan AWS SDK. Untuk menggunakan contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Java

Example — Hapus hibah akses

```
public void deleteAccessGrant() {
DeleteAccessGrantRequest deleteRequest = DeleteAccessGrantRequest.builder()
.accountId("111122223333")
.accessGrantId("a1b2c3d4-5678-90ab-cdef-EXAMPLE11111")
.build();
DeleteAccessGrantResponse deleteResponse =
 s3Control.deleteAccessGrant(deleteRequest);
LOGGER.info("DeleteAccessGrantResponse: " + deleteResponse);
}
```

Respons:

```
DeleteAccessGrantResponse()
```

Minta akses ke data Amazon S3 melalui S3 Access Grants

Setelah Anda menggunakan Hibah Akses Amazon S3 untuk membuat hibah akses yang memberikan kepala sekolah AWS Identity and Access Management (IAM), identitas direktori perusahaan Anda, atau akses aplikasi resmi ke data S3 Anda, penerima hibah Anda dapat meminta kredensyal untuk mengakses data ini.

Saat aplikasi atau Layanan AWS menggunakan operasi `GetDataAccess` API untuk meminta S3 Access Grants untuk mengakses data S3 Anda atas nama penerima hibah, S3 Access Grants terlebih dahulu memverifikasi bahwa Anda telah memberikan akses identitas ini ke data. Kemudian, S3 Access Grants menggunakan operasi AssumeRoleAPI untuk mendapatkan token kredensyal sementara dan menjualnya ke pemohon. Token kredensial sementara ini adalah token AWS Security Token Service (AWS STS).

Permintaan `GetDataAccess` harus menyertakan parameter `target`, yang menentukan ruang lingkup data S3 yang berlaku untuk kredensial sementara. Ruang lingkup `target` ini dapat sama

dengan ruang lingkup pemberian atau bagian dari ruang lingkup itu, tetapi ruang lingkup `target` harus dalam ruang lingkup pemberian yang diberikan kepada pemohon. Permintaan juga harus menentukan `permission` parameter untuk menunjukkan tingkat izin untuk kredensial sementara, apakah `READ`, `WRITE`, atau `READWRITE`.

Pemohon dapat menentukan tingkat hak istimewa token sementara dalam permintaan kredensialnya. Dengan menggunakan parameter `privilege`, pemohon dapat mengurangi atau meningkatkan ruang lingkup akses kredensial sementara, dalam batas-batas ruang lingkup pemberian. Nilai default `privilege` parameter adalah`Default`, yang berarti bahwa cakupan target dari kredensi yang dikembalikan adalah cakupan hibah asli. Nilai lain yang mungkin untuk `privilege` adalah `Minimal`. Jika ruang lingkup `target` dikurangi dari ruang lingkup pemberian asli, maka kredensial sementara dihilangkan cakupannya agar sesuai dengan ruang lingkup `target`, selama ruang lingkup berada dalam ruang lingkup `target` pemberian.

Tabel berikut merinci efek parameter `privilege` pada dua pemberian. Satu pemberian memiliki ruang lingkup `S3://`*`example-s3-bucket1`*`/bob/*`, yang mencakup seluruh `bob/` prefiks dalam bucket *`example-s3-bucket1`*. Hibah lainnya memiliki ruang lingkup `S3://`*`example-s3-bucket1`*`/bob/reports/*`, yang hanya mencakup `bob/reports/` prefiks di dalam bucket *`example-s3-bucket1`*.

Lingkup pemberian	Lingkup yang diminta	Keistimewaan	Lingkup yang dikembalikan	Efek
`S3://`*`example-s3-bucket1`*`/bob/*`	*`example-s3-bucket1`*`/bob/*`	`Default`	*`example-s3-bucket1`*`/bob/*`	Pemohon memiliki akses ke semua objek yang memiliki nama kunci yang dimulai dengan prefiks *`bob/`* di dalam bucket *`example-s3-bucket1`* .
`S3://`*`example-s3-bucket1`*`/bob/*`	*`example-s3-bucket1`*`/bob/`	`Minimal`	*`example-s3-bucket1`*`/bob/`	Tanpa karakter wild card* setelah nama prefiksbob/, pemohon hanya memiliki akses ke objek yang disebutkan bob/ dalam bucket

Lingkup pemberian	Lingkup yang diminta	Keistimewaan	Lingkup yang dikembalikan	Efek
				example-s3-bucket1 . Tidak umum memiliki objek seperti itu. Pemohon tidak memiliki akses ke objek lain, termasuk yang memiliki nama kunci yang dimulai dengan prefiks bob/.
S3://*example-s3-bucket1* /bob/*	*example-s3-bucket1* /bob/images/*	Minimal	*example-s3-bucket1* /bob/images/*	Pemohon memiliki akses ke semua objek yang memiliki nama kunci yang dimulai dengan prefiks *bob/images/** di dalam bucket *example-s3-bucket1* .
S3://*example-s3-bucket1* /bob/reports/*	*example-s3-bucket1* /bob/reports/file.txt	Default	*example-s3-bucket1* /bob/reports/*	Pemohon memiliki akses ke semua objek yang memiliki nama kunci yang dimulai dengan prefiks bob/reports di dalam bucket *example-s3-bucket1* , yang merupakan cakupan pemberian yang cocok.

Lingkup pemberian	Lingkup yang diminta	Keistimewaan	Lingkup yang dikembalikan	Efek
S3://*example-s3-bucket1* /bob/reports/*	*example-s3-bucket1* /bob/reports/file.txt	Minimal	*example-s3-bucket1* /bob/reports/file.txt	Pemohon hanya memiliki akses ke objek dengan nama kunci bob/reports/file.txt di dalam bucket *example-s3-bucket1* . Pemohon tidak memiliki akses ke objek lain.

Parameter `durationSeconds` menetapkan durasi kredensial sementara, dalam hitungan detik. Nilai default adalah `3600` detik (1 jam), tetapi pemohon (penerima pemberian) dapat menentukan rentang dari `900` detik (15 menit) hingga `43200` detik (12 jam). Jika penerima pemberian meminta nilai yang lebih tinggi dari maksimum ini, permintaan gagal.

> **Note**
>
> Dalam permintaan Anda untuk token sementara, jika lokasi adalah objek, tetapkan nilai parameter `targetType` dalam permintaan `Object` Anda. Parameter ini diperlukan hanya jika lokasi adalah objek dan tingkat hak istimewa adalah `Minimal`. Jika lokasi adalah bucket atau prefiks, Anda tidak perlu menentukan parameter ini.

Untuk informasi selengkapnya, lihat GetDataAccessdi Referensi API Amazon Simple Storage Service.

Anda dapat meminta kredensyal sementara dengan menggunakan AWS Command Line Interface (AWS CLI), Amazon S3 REST API, dan SDK. AWS

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Example Meminta kredensial sementara

Permintaan:

```
aws s3control get-data-access \
--account-id 111122223333 \
--target s3://example-s3-bucket/prefixA* \
--permission READ \
--privilege Default \
--region us-east-2
```

Respons:

```
{
"Credentials": {
 "AccessKeyId": "Example-key-id",
 "SecretAccessKey": "Example-access-key",
 "SessionToken": "Example-session-token",
"Expiration": "2023-06-14T18:56:45+00:00"},
    "MatchedGrantTarget": "s3://example-s3-bucket/prefixA**"
}
```

Penggunaan API REST

Untuk informasi tentang dukungan Amazon S3 REST API untuk meminta kredensyal sementara dari S3 Access Grants, lihat GetDataAccessdi Referensi API Amazon Simple Storage Service.

Menggunakan AWS SDK

Bagian ini memberikan contoh bagaimana penerima hibah meminta kredensyal sementara dari S3 Access Grants dengan menggunakan SDK. AWS

Java

Contoh kode berikut mengembalikan kredensial sementara yang digunakan penerima pemberian untuk mengakses data S3 Anda. Untuk menggunakan contoh kode ini, ganti *user input placeholders* dengan informasi Anda sendiri.

Example Dapatkan kredensial sementara

Permintaan:

```
public void getDataAccess() {
GetDataAccessRequest getDataAccessRequest = GetDataAccessRequest.builder()
.accountId("111122223333")
.permission(Permission.READ)
.privilege(Privilege.MINIMAL)
.target("s3://example-s3-bucket/prefixA*")
.build();
GetDataAccessResponse getDataAccessResponse =
 s3Control.getDataAccess(getDataAccessRequest);
LOGGER.info("GetDataAccessResponse: " + getDataAccessResponse);
}
```

Respons:

```
GetDataAccessResponse(
Credentials=Credentials(
AccessKeyId="Example-access-key-id",
SecretAccessKey="Example-secret-access-key",
SessionToken="Example-session-token",
Expiration=2023-06-07T06:55:24Z
))
```

Akses data S3 melalui hibah akses

Setelah penerima hibah memperoleh kredensyal sementara melalui hibah akses mereka, mereka dapat menggunakan kredensyal sementara ini untuk memanggil operasi Amazon S3 API untuk mengakses data Anda.

Penerima hibah dapat mengakses data S3 dengan menggunakan AWS Command Line Interface (AWS CLI), AWS SDK, dan Amazon S3 REST API.

Menggunakan AWS CLI

Setelah penerima hibah memperoleh kredensyal sementara mereka dari S3 Access Grants, mereka dapat mengatur profil dengan kredensyal ini untuk mengambil data.

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Example — Mengatur profil

```
aws configure set aws_access_key_id "$accessKey" --profile access-grants-consumer-access-profile
aws configure set aws_secret_access_key "$secretKey" --profile access-grants-consumer-access-profile
aws configure set aws_session_token "$sessionToken" --profile access-grants-consumer-access-profile
```

Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Example — Dapatkan data S3

Penerima hibah dapat menggunakan get-object AWS CLI perintah untuk mengakses data. Penerima hibah juga dapat menggunakan put-object, ls, dan perintah AWS CLI S3 lainnya.

```
aws s3api get-object \
--bucket example-s3-bucket1 \
--key myprefix \
--region us-east-2 \
--profile access-grants-consumer-access-profile
```

Menggunakan AWS SDK

Bagian ini memberikan contoh bagaimana penerima hibah dapat mengakses data S3 Anda dengan menggunakan SDK. AWS

Java

> Untuk contoh cara mendapatkan data S3 dengan menggunakan kredensyal sementara, lihat cara mendapatkan objek dengan menggunakan contoh kode AWS SDK dan Amazon S3 untuk file. AWS SDK for Java 2.x

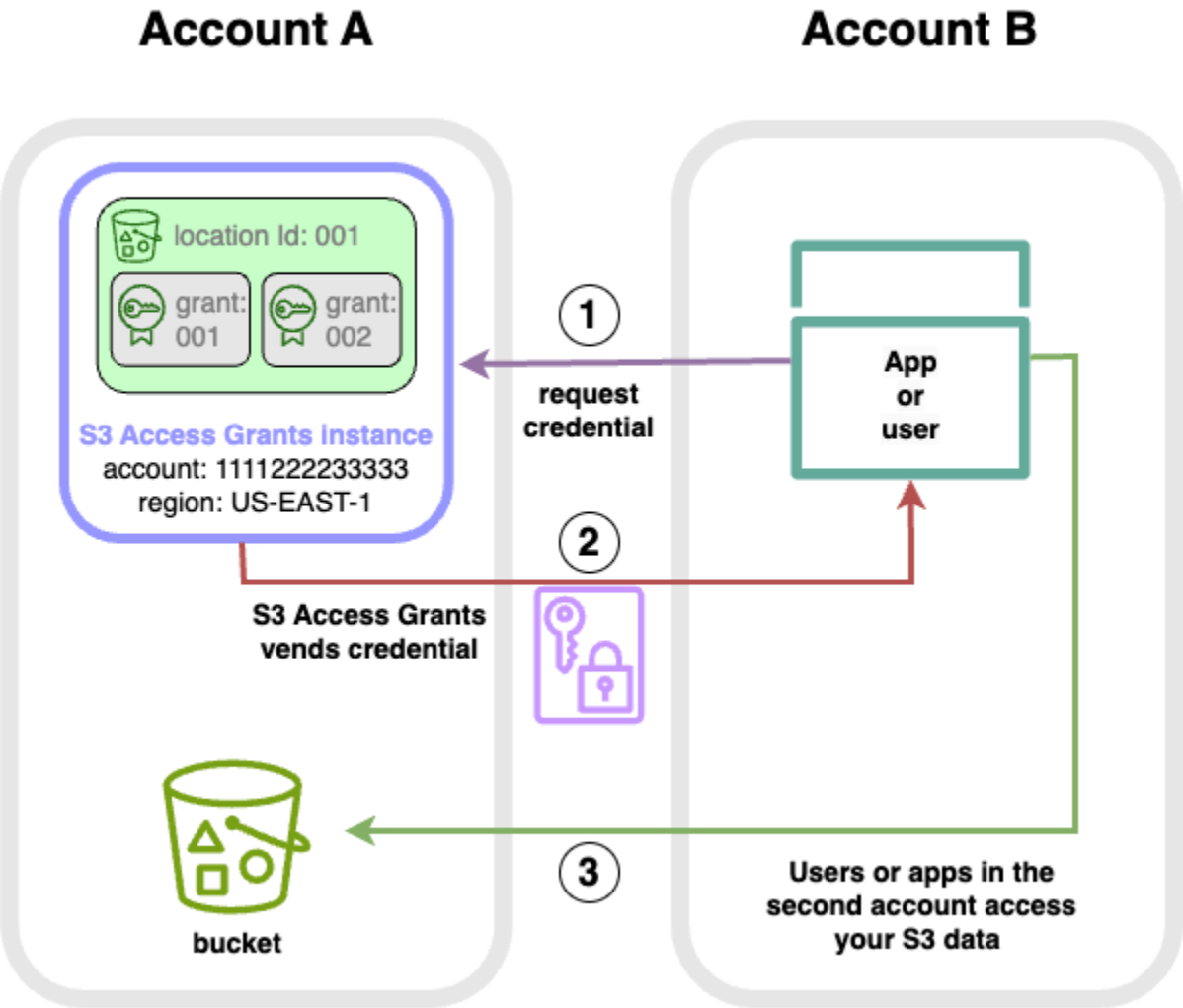
Akses S3 memberikan akses lintas akun

Dengan S3 Access Grants, Anda dapat memberikan akses data Amazon S3 ke hal-hal berikut:

- AWS Identity and Access Management (IAM) identitas dalam akun Anda
- Identitas IAM di akun lain AWS
- Pengguna direktori atau grup dalam AWS IAM Identity Center instans Anda

Pertama, konfigurasikan akses lintas akun untuk akun lain. Ini termasuk memberikan akses ke instans S3 Access Grants Anda dengan menggunakan kebijakan sumber daya. Kemudian, berikan akses ke data S3 Anda (bucket, awalan, atau objek) dengan menggunakan hibah.

Setelah Anda mengonfigurasi akses lintas akun, akun lain dapat meminta kredensyal akses sementara ke data Amazon S3 Anda dari S3 Access Grants. Gambar berikut menunjukkan alur pengguna untuk akses S3 lintas akun melalui S3 Access Grants:



1. Pengguna atau aplikasi di akun kedua (B) meminta kredensyal dari instans S3 Access Grants di akun Anda (A), tempat data Amazon S3 disimpan. Untuk informasi selengkapnya, lihat Minta akses ke data Amazon S3 melalui S3 Access Grants.
2. Instans S3 Access Grants di akun Anda (A) mengembalikan kredensyal sementara jika ada hibah yang memberikan akses akun kedua ke data Amazon S3 Anda. Untuk informasi selengkapnya, lihat the section called “Membuat pemberian”.
3. Pengguna atau aplikasi di akun kedua (B) menggunakan kredensyal yang dijual oleh S3 Access Grants untuk mengakses data S3 di akun Anda (A).

Mengkonfigurasi Akses S3 Memberikan akses lintas akun

Untuk memberikan akses S3 lintas akun melalui S3 Access Grants, ikuti langkah-langkah berikut:

- Langkah 1: Konfigurasikan instance S3 Access Grants di akun Anda, misalnya, ID akun`111122223333`, tempat data S3 disimpan.
- Langkah 2: Konfigurasikan kebijakan sumber daya untuk instans S3 Access Grants di akun Anda `111122223333` untuk memberikan akses ke akun kedua, misalnya, ID akun. `444455556666`
- Langkah 3: Konfigurasikan izin IAM untuk IAM Principal di akun kedua `444455556666` untuk meminta kredensyal dari instans S3 Access Grants di akun Anda. `111122223333`
- Langkah 4: Buat hibah di akun Anda `111122223333` yang memberikan Kepala Sekolah IAM di akun kedua `444455556666` akses ke beberapa data S3 di akun Anda. `111122223333`

Langkah 1: Konfigurasikan instance S3 Access Grants di akun Anda

Pertama, Anda harus memiliki instans S3 Access Grants di akun Anda `111122223333` untuk mengelola akses ke data Amazon S3 Anda. Anda harus membuat instance S3 Access Grants di setiap Wilayah AWS tempat data S3 yang ingin Anda bagikan disimpan. Jika Anda berbagi data di lebih dari satu Wilayah AWS, ulangi setiap langkah konfigurasi ini untuk masing-masing Wilayah AWS. Jika Anda sudah memiliki instance S3 Access Grants di Wilayah AWS tempat penyimpanan data S3 Anda, lanjutkan ke langkah berikutnya. Jika Anda belum mengonfigurasi instance S3 Access Grants, lihat Membuat instans S3 Access Grants untuk menyelesaikan langkah ini.

Langkah 2: Konfigurasikan kebijakan sumber daya untuk instans S3 Access Grants Anda untuk memberikan akses lintas akun

Setelah Anda membuat instans S3 Access Grants di akun Anda `111122223333` untuk akses lintas akun, konfigurasikan kebijakan berbasis sumber daya untuk instans S3 Access Grants di akun Anda

untuk memberikan akses lintas akun. `111122223333` Instans S3 Access Grants sendiri mendukung kebijakan berbasis sumber daya. Dengan kebijakan berbasis sumber daya yang benar, Anda dapat memberikan akses untuk pengguna AWS Identity and Access Management (IAM) atau peran dari orang lain Akun AWS ke instans Hibah Akses S3 Anda. Akses lintas akun hanya memberikan izin (tindakan) ini:

- `s3:GetAccessGrantsInstanceForPrefix`— pengguna, peran, atau aplikasi dapat mengambil instance S3 Access Grants yang berisi awalan tertentu.
- `s3:ListAccessGrants`
- `s3:ListAccessLocations`
- `s3:GetDataAccess`— pengguna, peran, atau aplikasi dapat meminta kredensyal sementara berdasarkan akses yang diberikan kepada Anda melalui Hibah Akses S3. Gunakan kredensial ini untuk mengakses data S3 yang telah diberikan akses kepada Anda.

Anda dapat memilih izin mana yang akan disertakan dalam kebijakan sumber daya. Kebijakan sumber daya pada instans S3 Access Grants ini adalah kebijakan berbasis sumber daya normal dan mendukung semua yang didukung oleh bahasa kebijakan IAM. Dalam kebijakan yang sama, Anda dapat memberikan akses ke identitas IAM tertentu di akun Anda`111122223333`, misalnya, dengan menggunakan `aws:PrincipalArn` kondisi, tetapi Anda tidak harus melakukannya dengan Hibah Akses S3. Sebagai gantinya, dalam instance S3 Access Grants Anda, Anda dapat membuat hibah untuk identitas IAM individual dari akun Anda, serta untuk akun lainnya. Dengan mengelola setiap hibah akses melalui S3 Access Grants, Anda dapat menskalakan izin Anda.

Jika Anda sudah menggunakan AWS Resource Access Manager(AWS RAM), Anda dapat menggunakannya untuk membagikan `s3:AccessGrants`sumber daya Anda dengan akun lain atau di dalam organisasi Anda. Lihat Bekerja dengan AWS sumber daya bersama untuk informasi selengkapnya. Jika tidak digunakan AWS RAM, Anda juga dapat menambahkan kebijakan sumber daya dengan menggunakan operasi S3 Access Grants API atau AWS Command Line Interface ()AWS CLI.

Menggunakan konsol S3

Kami menyarankan Anda menggunakan AWS Resource Access Manager (AWS RAM) Konsol untuk membagikan `s3:AccessGrants` sumber daya Anda dengan akun lain atau di dalam organisasi Anda. Untuk membagikan akun lintas akun S3 Access Grants, lakukan hal berikut:

Untuk mengonfigurasi kebijakan sumber daya instans S3 Access Grants:

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pilih Wilayah AWS dari Wilayah AWS pemilih.
3. Dari panel navigasi kiri, pilih Access Grants.
4. Pada halaman instance Access Grants, di bagian Instans di akun ini, pilih Bagikan instance. Ini akan mengarahkan Anda ke AWS RAM Konsol.
5. Pilih Buat berbagi sumber daya.
6. Ikuti AWS RAM langkah-langkah untuk membuat pembagian sumber daya. Untuk informasi selengkapnya, lihat Membuat pembagian sumber daya di AWS RAM.

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Anda dapat menambahkan kebijakan sumber daya dengan menggunakan perintah `put-access-grants-instance-resource-policy` CLI.

Jika Anda ingin memberikan akses lintas akun untuk instans S3 Access Grants ada di akun Anda `111122223333` ke akun kedua`444455556666`, kebijakan sumber daya untuk instance S3 Access Grants di akun Anda `111122223333` harus memberikan `444455556666` izin akun kedua untuk melakukan tindakan berikut:

- `s3:ListAccessGrants`
- `s3:ListAccessGrantsLocations`
- `s3:GetDataAccess`
- `s3:GetAccessGrantsInstanceForPrefix`

Dalam kebijakan sumber daya instans S3 Access Grants, tentukan ARN instance S3 Access Grants Anda sebagai`Resource`, dan akun kedua sebagai. `444455556666` `Principal` Untuk menggunakan contoh berikut, ganti *`placeholder input pengguna dengan informasi`* Anda sendiri.

```
{
"Version": "2012-10-17",
```

```
"Statement": [
{
 "Effect": "Allow",
 "Principal": {
 "AWS": "444455556666"
},
 "Action": [
  "s3:ListAccessGrants",
  "s3:ListAccessGrantsLocations",
  "s3:GetDataAccess",
  "s3:GetAccessGrantsInstanceForPrefix"
 ],
 "Resource": "arn:aws:s3:us-east-2:111122223333:access-grants/default"
} ]
}
```

Untuk menambahkan atau memperbarui kebijakan sumber daya instans S3 Access Grants, gunakan perintah berikut. Bila Anda menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Example Menambahkan atau memperbarui kebijakan sumber daya instans S3 Access Grants

```
aws s3control put-access-grants-instance-resource-policy \
--account-id 111122223333 \
--policy file://resourcePolicy.json \
--region us-east-2
{
 "Policy": "{\n
   \"Version\": \"2012-10-17\",\n
   \"Statement\": [{\n
  \"Effect\": \"Allow\",\n
  \"Principal\": {\n
    \"AWS\": \"444455556666\"\n
  },\n
  \"Action\": [\n
    \"s3:ListAccessGrants\",\n
    \"s3:ListAccessGrantsLocations\",\n
    \"s3:GetDataAccess\",\n
    \"s3:GetAccessGrantsInstanceForPrefix\"\n
  ],\n
  \"Resource\": \"arn:aws:s3:us-east-2:111122223333:access-grants/default\"\n
    }\n
```

```
    ]\n
    }\n",
  "CreatedAt": "2023-06-16T00:07:47.473000+00:00"
 }
```

Example Mendapatkan kebijakan sumber daya S3 Access Grants

Anda juga dapat menggunakan CLI untuk mendapatkan atau menghapus kebijakan sumber daya untuk instance S3 Access Grants.

Untuk mendapatkan kebijakan sumber daya S3 Access Grants, gunakan perintah contoh berikut. Untuk menggunakan contoh perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control get-access-grants-instance-resource-policy \
--account-id 111122223333 \
--region us-east-2

{
"Policy": "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow\",\"Principal
\":{\"AWS\":\"arn:aws:iam::111122223333:root\"},\"Action\":[\"s3:ListAccessGrants\",
\"s3:ListAccessGrantsLocations\",\"s3:GetDataAccess\"],\"Resource\":\"arn:aws:s3:us-
east-2:111122223333:access-grants/default\"}]}",
"CreatedAt": "2023-06-16T00:07:47.473000+00:00"
}
```

Example Menghapus kebijakan sumber daya S3 Access Grants

Untuk menghapus kebijakan sumber daya S3 Access Grants, gunakan perintah contoh berikut. Untuk menggunakan contoh perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control delete-access-grants-instance-resource-policy \
--account-id 111122223333 \
--region us-east-2

// No response body
```

Penggunaan API REST

Anda dapat menambahkan kebijakan sumber daya dengan menggunakan PutAccessGrantsInstanceResourcePolicy API.

Jika Anda ingin memberikan akses lintas akun untuk instans S3 Access Grants ada di akun Anda `111122223333` ke akun kedua`444455556666`, kebijakan sumber daya untuk instance S3 Access Grants di akun Anda `111122223333` harus memberikan `444455556666` izin akun kedua untuk melakukan tindakan berikut:

- `s3:ListAccessGrants`
- `s3:ListAccessGrantsLocations`
- `s3:GetDataAccess`
- `s3:GetAccessGrantsInstanceForPrefix`

Dalam kebijakan sumber daya instans S3 Access Grants, tentukan ARN instance S3 Access Grants Anda sebagai`Resource`, dan akun kedua sebagai. `444455556666` `Principal` Untuk menggunakan contoh berikut, ganti *`placeholder input pengguna dengan informasi`* Anda sendiri.

```
{
"Version": "2012-10-17",
"Statement": [
{
 "Effect": "Allow",
 "Principal": {
 "AWS": "444455556666"
},
 "Action": [
  "s3:ListAccessGrants",
  "s3:ListAccessGrantsLocations",
  "s3:GetDataAccess",
  "s3:GetAccessGrantsInstanceForPrefix"
 ],
 "Resource": "arn:aws:s3:us-east-2:111122223333:access-grants/default"
} ]
}
```

Anda kemudian dapat menggunakan PutAccessGrantsInstanceResourcePolicy API untuk mengonfigurasi kebijakan.

Untuk informasi tentang dukungan REST API untuk memperbarui, mendapatkan, atau menghapus kebijakan sumber daya untuk instance S3 Access Grants, lihat bagian berikut di Referensi API Amazon Simple Storage Service:

- PutAccessGrantsInstanceResourcePolicy
- GetAccessGrantsInstanceResourcePolicy
- DeleteAccessGrantsInstanceResourcePolicy

Menggunakan AWS SDK

Bagian ini memberi Anda contoh AWS SDK tentang cara mengonfigurasi kebijakan sumber daya S3 Access Grants Anda untuk memberikan akses AWS akun kedua ke beberapa data S3 Anda.

Java

Tambahkan, perbarui, dapatkan, atau hapus kebijakan sumber daya untuk mengelola akses lintas akun ke instans S3 Access Grants Anda.

Example Menambahkan atau memperbarui kebijakan sumber daya instans S3 Access Grants

Jika Anda ingin memberikan akses lintas akun untuk instans S3 Access Grants ada di akun Anda `111122223333` ke akun kedua`444455556666`, kebijakan sumber daya untuk instance S3 Access Grants di akun Anda `111122223333` harus memberikan `444455556666` izin akun kedua untuk melakukan tindakan berikut:

- `s3:ListAccessGrants`
- `s3:ListAccessGrantsLocations`
- `s3:GetDataAccess`
- `s3:GetAccessGrantsInstanceForPrefix`

Dalam kebijakan sumber daya instans S3 Access Grants, tentukan ARN instance S3 Access Grants Anda sebagai`Resource`, dan akun kedua sebagai. `444455556666` `Principal` Untuk menggunakan contoh berikut, ganti *`placeholder input pengguna dengan informasi`* Anda sendiri.

```
{
"Version": "2012-10-17",
"Statement": [
{
 "Effect": "Allow",
 "Principal": {
 "AWS": "444455556666"
```

```
},
 "Action": [
  "s3:ListAccessGrants",
  "s3:ListAccessGrantsLocations",
  "s3:GetDataAccess",
  "s3:GetAccessGrantsInstanceForPrefix"
 ],
 "Resource": "arn:aws:s3:us-east-2:111122223333:access-grants/default"
} ]
}
```

Untuk menambah atau memperbarui kebijakan sumber daya instans S3 Access Grants, gunakan contoh kode berikut:

```
public void putAccessGrantsInstanceResourcePolicy() {
 PutAccessGrantsInstanceResourcePolicyRequest putRequest =
 PutAccessGrantsInstanceResourcePolicyRequest.builder()
 .accountId(111122223333)
 .policy(RESOURCE_POLICY)
 .build();
 PutAccessGrantsInstanceResourcePolicyResponse putResponse =
 s3Control.putAccessGrantsInstanceResourcePolicy(putRequest);
 LOGGER.info("PutAccessGrantsInstanceResourcePolicyResponse: " + putResponse);
 }
```

Respons:

```
PutAccessGrantsInstanceResourcePolicyResponse(
 Policy={
 "Version": "2012-10-17",
 "Statement": [{
 "Effect": "Allow",
 "Principal": {
 "AWS": "444455556666"
 },
 "Action": [
 "s3:ListAccessGrants",
 "s3:ListAccessGrantsLocations",
 "s3:GetDataAccess",
 "s3:GetAccessGrantsInstanceForPrefix"
 ],
 "Resource": "arn:aws:s3:us-east-2:111122223333:access-grants/default"
 }]
```

```
 }
 )
```

Example Mendapatkan kebijakan sumber daya S3 Access Grants

Untuk mendapatkan kebijakan sumber daya S3 Access Grants, gunakan contoh kode berikut. Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

```
public void getAccessGrantsInstanceResourcePolicy() {
 GetAccessGrantsInstanceResourcePolicyRequest getRequest =
 GetAccessGrantsInstanceResourcePolicyRequest.builder()
 .accountId(111122223333)
 .build();
 GetAccessGrantsInstanceResourcePolicyResponse getResponse =
 s3Control.getAccessGrantsInstanceResourcePolicy(getRequest);
 LOGGER.info("GetAccessGrantsInstanceResourcePolicyResponse: " + getResponse);
 }
```

Respons:

```
GetAccessGrantsInstanceResourcePolicyResponse(
 Policy={"Version":"2012-10-17","Statement":[{"Effect":"Allow","Principal":
{"AWS":"arn:aws:iam::444455556666:root"},"Action":
["s3:ListAccessGrants","s3:ListAccessGrantsLocations","s3:GetDataAccess"],"Resource":"arn:aw
east-2:111122223333:access-grants/default"}]},
 CreatedAt=2023-06-15T22:54:44.319Z
 )
```

Example Menghapus kebijakan sumber daya S3 Access Grants

Untuk menghapus kebijakan sumber daya S3 Access Grants, gunakan contoh kode berikut. Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

```
public void deleteAccessGrantsInstanceResourcePolicy() {
 DeleteAccessGrantsInstanceResourcePolicyRequest deleteRequest =
 DeleteAccessGrantsInstanceResourcePolicyRequest.builder()
 .accountId(111122223333)
 .build();
 DeleteAccessGrantsInstanceResourcePolicyResponse deleteResponse =
 s3Control.putAccessGrantsInstanceResourcePolicy(deleteRequest);
```

```
LOGGER.info("DeleteAccessGrantsInstanceResourcePolicyResponse: " + deleteResponse);
}
```

Respons:

```
DeleteAccessGrantsInstanceResourcePolicyResponse()
```

Langkah 3: Berikan identitas IAM di izin akun kedua untuk memanggil instans S3 Access Grants di akun Anda

Setelah pemilik data Amazon S3 mengonfigurasi kebijakan lintas akun untuk instans S3 Access Grants di akun`111122223333`, pemilik akun kedua `444455556666` harus membuat kebijakan berbasis identitas untuk pengguna atau peran IAM-nya, dan pemilik harus memberi mereka akses ke instans S3 Access Grants. Dalam kebijakan berbasis identitas, sertakan satu atau beberapa tindakan berikut, tergantung pada apa yang diberikan dalam kebijakan sumber daya instans S3 Access Grants dan izin yang ingin Anda berikan:

- `s3:ListAccessGrants`
- `s3:ListAccessGrantsLocations`
- `s3:GetDataAccess`
- `s3:GetAccessGrantsInstanceForPrefix`

Mengikuti pola akses AWS lintas akun, pengguna IAM atau peran di akun kedua `444455556666` harus secara eksplisit memiliki satu atau lebih izin ini. Misalnya, berikan `s3:GetDataAccess` izin agar pengguna atau peran IAM dapat memanggil instans S3 Access Grants di akun `111122223333` untuk meminta kredensyal.

Untuk menggunakan contoh perintah ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
  "Effect": "Allow",
  "Action": [
   "s3:GetDataAccess",
```

```
  ],
   "Resource": "arn:aws:s3:us-east-2:111122223333:access-grants/default"
  }
 ]
}
```

Untuk informasi tentang mengedit kebijakan berbasis identitas IAM, lihat Mengedit kebijakan IAM dalam panduan.AWS Identity and Access Management

Langkah 4: Buat hibah di instance S3 Access Grants dari akun Anda yang memberikan identitas IAM di akun kedua akses ke beberapa data S3 Anda

Untuk langkah konfigurasi terakhir, Anda dapat membuat hibah di instans S3 Access Grants di akun 111122223333 yang memberikan akses ke identitas IAM di akun kedua 444455556666 ke beberapa data S3 di akun Anda. Anda dapat melakukannya dengan menggunakan Konsol Amazon S3, CLI, API, dan SDK. Untuk informasi selengkapnya, lihat Membuat pemberian.

Dalam hibah, tentukan AWS ARN identitas IAM dari akun kedua, dan tentukan lokasi mana dalam data S3 Anda (ember, awalan, atau objek) yang Anda berikan akses. Lokasi ini harus sudah terdaftar dengan instans S3 Access Grants Anda. Untuk informasi selengkapnya, lihat Mendaftarkan lokasi. Anda dapat secara opsional menentukan subprefix. Misalnya, jika lokasi yang Anda berikan akses adalah bucket, dan Anda ingin membatasi akses lebih jauh ke objek tertentu di bucket tersebut, maka berikan nama kunci objek di `S3SubPrefix` bidang tersebut. Atau jika Anda ingin membatasi akses ke objek di bucket dengan nama kunci yang dimulai dengan awalan tertentu, seperti`2024-03-research-results/`, maka teruskan`S3SubPrefix=2024-03-research-results/`.

Berikut ini adalah contoh perintah CLI untuk membuat hibah akses untuk identitas di akun kedua. Untuk informasi selengkapnya, lihat Membuat pemberian. Untuk menggunakan contoh perintah ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control create-access-grant \
--account-id 111122223333 \
--access-grants-location-id default \
--access-grants-location-configuration S3SubPrefix=prefixA* \
--permission READ \
--grantee GranteeType=IAM,GranteeIdentifier=arn:aws:iam::444455556666:role/data-consumer-1
```

Setelah mengonfigurasi akses lintas akun, pengguna atau peran di akun kedua dapat melakukan hal berikut:

- Panggilan `ListAccessGrantsInstances` untuk mencantumkan instance S3 Access Grants yang dibagikan dengannya. AWS RAM Untuk informasi selengkapnya, lihat Lihat detail instance S3 Access Grants.
- Meminta kredensi sementara dari S3 Access Grants. Untuk informasi selengkapnya tentang cara membuat permintaan ini, lihatMinta akses ke data Amazon S3 melalui S3 Access Grants.

Menggunakan AWS tag dengan S3 Access Grants

Tag di Amazon S3 Access Grants memiliki karakteristik yang mirip dengan tag objek di Amazon S3. Setiap tag adalah pasangan nilai kunci. Sumber daya di S3 Access Grants yang dapat Anda tag adalah instans, lokasi, dan pemberian S3 Access Grants.

> **Note**
>
> Penandaan di S3 Access Grants menggunakan operasi API yang berbeda dari penandaan objek. S3 Access Grants menggunakan operasi TagResource, UntagResource, dan ListTagsForResourceAPI, di mana sumber daya dapat berupa instance S3 Access Grants, lokasi terdaftar, atau hibah akses.

Mirip dengan tag objek, batasan berikut berlaku:

- Anda dapat menambahkan tag ke sumber daya baru S3 Access Grants saat Anda membuatnya, atau Anda dapat menambahkan tag ke sumber daya yang sudah ada.
- Anda dapat mengaitkan hingga 10 tag dengan sumber daya. Jika beberapa tag terkait dengan sumber daya yang sama, mereka harus memiliki kunci tag unik.
- Kunci tanda dapat terdiri dari hingga 128 karakter Unicode, dan nilai tanda dapat terdiri dari hingga 256 karakter Unicode. Tag diwakili secara internal dalam UTF-16. Dalam UTF-16, karakter mengkonsumsi 1 atau 2 posisi karakter.
- Kunci dan nilainya peka huruf besar/kecil.

Untuk informasi selengkapnya tentang pembatasan tag, lihat Pembatasan tag yang ditentukan pengguna di AWS Billing Panduan Pengguna.

Anda dapat menandai resource di S3 Access Grants dengan menggunakan AWS Command Line Interface (AWS CLI), Amazon S3 REST API, atau SDK. AWS

Menggunakan AWS CLI

Untuk menginstal AWS CLI, lihat Menginstal AWS CLI di Panduan AWS Command Line Interface Pengguna.

Anda dapat menandai sumber daya S3 Access Grants saat Anda membuatnya atau setelah Anda membuatnya. Contoh berikut menunjukkan bagaimana Anda menandai atau menghapus tag instans S3 Access Grants. Anda dapat melakukan operasi serupa untuk lokasi terdaftar dan pemberian akses.

Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

Example — Buat instance S3 Access Grants dengan tag

```
aws s3control create-access-grants-instance \
 --account-id 111122223333 \
 --profile access-grants-profile \
 --region us-east-2 \
 --tags Key=tagKey1,Value=tagValue1
```

Respons:

```
 {
    "CreatedAt": "2023-10-25T01:09:46.719000+00:00",
    "AccessGrantsInstanceId": "default",
    "AccessGrantsInstanceArn": "arn:aws:s3:us-east-2:111122223333:access-grants/
default"
}
```

Example — Tag instance S3 Access Grants yang sudah dibuat

```
aws s3control tag-resource \
--account-id 111122223333 \
--resource-arn "arn:aws:s3:us-east-2:111122223333:access-grants/default" \
--profile access-grants-profile \
--region us-east-2 \
--tags Key=tagKey2,Value=tagValue2
```

Example — Daftar tag untuk instance S3 Access Grants

```
aws s3control list-tags-for-resource \
```

```
--account-id 111122223333 \
--resource-arn "arn:aws:s3:us-east-2:111122223333:access-grants/default" \
--profile access-grants-profile \
--region us-east-2
```

Respons:

```
{
    "Tags": [
        {
            "Key": "tagKey1",
            "Value": "tagValue1"
        },
        {
            "Key": "tagKey2",
            "Value": "tagValue2"
        }
    ]
}
```

Example — Hapus tag instance Hibah Akses S3

```
aws s3control untag-resource \
 --account-id 111122223333 \
 --resource-arn "arn:aws:s3:us-east-2:111122223333:access-grants/default" \
 --profile access-grants-profile \
 --region us-east-2 \
 --tag-keys "tagKey2"
```

Penggunaan API REST

Anda dapat menggunakan Amazon S3 API untuk menandai, menghapus tag, atau mencantumkan tag untuk instance S3 Access Grants, lokasi terdaftar, atau hibah akses. Untuk informasi tentang dukungan REST API untuk mengelola tag S3 Access Grants, lihat bagian berikut di Referensi API Amazon Simple Storage Service:

- TagResource
- UntagResource
- ListTagsForResource

Batasan S3 Access Grants

S3 Access Grants memiliki keterbatasan sebagai berikut:

> **Note**
>
> Jika kasus penggunaan Anda melebihi batasan ini, hubungi AWS dukungan untuk meminta batas yang lebih tinggi.

Instans S3 Access Grants

Anda dapat membuat 1 instance Hibah Akses S3 Wilayah AWS per akun. Lihat Membuat instans S3 Access Grants.

Lokasi S3 Access Grants

Anda dapat mendaftarkan 1.000 lokasi S3 Access Grants per instans S3 Access Grants. Lihat Daftarkan lokasi S3 Access Grants.

Pemberian Izin

Anda dapat membuat 100.000 pemberian per instans S3 Access Grants. Lihat Membuat pemberian.

Integrasi S3 Access Grants

Hibah Akses S3 dapat digunakan dengan AWS layanan dan fitur berikut. Halaman ini akan diperbarui saat integrasi baru tersedia.

AWS IAM Identity Center

Propagasi identitas tepercaya di seluruh aplikasi

Amazon EMR

Luncurkan klaster Amazon EMR dengan S3 Access Grants

Amazon EMR di EKS

Luncurkan Amazon EMR di klaster EKS dengan S3 Access Grants

Aplikasi Amazon EMR Nirserver

- Luncurkan aplikasi Amazon EMR Nirserver dengan S3 Access Grants

Amazon Athena

- Menggunakan IAM Identity Center mengaktifkan kelompok kerja Athena,

Mengelola Akses dengan ACL

Daftar kontrol akses (ACL) adalah salah satu opsi berbasis sumber daya yang dapat Anda gunakan untuk mengelola akses ke bucket dan objek Anda. Anda dapat menggunakan ACL untuk memberikan izin baca/tulis dasar kepada yang lain. Akun AWS Ada batasan-batasan untuk mengelola izin dengan menggunakan ACL.

Misalnya, Anda hanya dapat memberikan izin kepada orang lain Akun AWS; Anda tidak dapat memberikan izin kepada pengguna di akun Anda. Anda tidak dapat memberikan izin bersyarat, atau Anda tidak dapat secara jelas menolak izin. ACL sesuai untuk skenario tertentu. Misalnya, jika pemilik bucket mengizinkan orang lain Akun AWS untuk mengunggah objek, izin ke objek ini hanya dapat dikelola menggunakan objek ACL oleh Akun AWS yang memiliki objek tersebut.

Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda, serta menonaktifkan atau mengaktifkan ACL. Secara default, Kepemilikan Objek diatur ke pengaturan yang diberlakukan pemilik Bucket, dan semua ACL dinonaktifkan. Ketika ACL dinonaktifkan, pemilik bucket memiliki semua objek di bucket dan mengelola akses ke objek-objek tersebut secara eksklusif dengan menggunakan kebijakan manajemen akses.

Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL. Kami menyarankan agar ACL dinonaktifkan, kecuali dalam keadaan yang tidak biasa ketika Anda perlu mengontrol akses untuk setiap objek secara individual. Dengan ACL dinonaktifkan, Anda dapat menggunakan kebijakan untuk mengontrol akses ke semua objek di bucket, terlepas dari siapa yang mengunggah objek ke bucket Anda. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

> **Important**
>
> Jika bucket Anda menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek S3, Anda harus menggunakan kebijakan untuk memberikan akses ke bucket Anda, serta objek di dalamnya. Dengan mengaktifkan pengaturan

yang diberlakukan pemilik Bucket, permintaan untuk mengatur daftar kontrol akses (ACL) atau memperbarui ACL akan gagal dan mengembalikan kode kesalahan `AccessControlListNotSupported`. Permintaan untuk membaca ACL masih didukung.

Untuk informasi selengkapnya tentang ACL, lihat topik berikut.

Topik

- Gambaran umum daftar kontrol akses (ACL)
- Mengonfigurasi ACL
- Contoh kebijakan untuk ACL

Gambaran umum daftar kontrol akses (ACL)

Daftar kontrol akses (ACL) Amazon S3 memungkinkan Anda mengelola akses ke bucket dan objek. Setiap bucket dan objek memiliki ACL yang melekat padanya sebagai sumber daya tambahan. Ini mendefinisikan Akun AWS atau kelompok mana yang diberikan akses dan jenis akses. Saat permintaan diterima terhadap sumber daya, Amazon S3 memeriksa ACL yang sesuai untuk memverifikasi bahwa pengirim permintaan memiliki izin akses yang diperlukan.

Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda, serta menonaktifkan atau mengaktifkan ACL. Secara default, Kepemilikan Objek diatur ke pengaturan yang diberlakukan pemilik Bucket, dan semua ACL dinonaktifkan. Ketika ACL dinonaktifkan, pemilik bucket memiliki semua objek di bucket dan mengelola akses ke objek-objek tersebut secara eksklusif dengan menggunakan kebijakan manajemen akses.

Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL. Kami menyarankan agar ACL dinonaktifkan, kecuali dalam keadaan yang tidak biasa ketika Anda perlu mengontrol akses untuk setiap objek secara individual. Dengan ACL dinonaktifkan, Anda dapat menggunakan kebijakan untuk mengontrol akses ke semua objek di bucket, terlepas dari siapa yang mengunggah objek ke bucket Anda. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

> **Important**
>
> Jika bucket Anda menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek S3, Anda harus menggunakan kebijakan untuk memberikan

akses ke bucket Anda, serta objek di dalamnya. Dengan mengaktifkan pengaturan yang diberlakukan pemilik Bucket, permintaan untuk mengatur daftar kontrol akses (ACL) atau memperbarui ACL akan gagal dan mengembalikan kode kesalahan `AccessControlListNotSupported`. Permintaan untuk membaca ACL masih didukung.

Saat Anda membuat bucket atau objek, Amazon S3 membuat ACL bawaan yang memberikan kendali penuh kepada pemilik sumber daya atas sumber daya tersebut. Hal ini ditunjukkan dalam ACL bucket sampel berikut (ACL objek bawaan memiliki struktur yang sama):

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Owner>
    <ID>*** Owner-Canonical-User-ID ***</ID>
    <DisplayName>owner-display-name</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:type="Canonical User">
        <ID>*** Owner-Canonical-User-ID ***</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
</AccessControlPolicy>
```

ACL sampel menyertakan elemen `Owner` yang mengidentifikasi pemilik dengan ID pengguna kanonik Akun AWS. Untuk petunjuk bagaimana menemukan id pengguna kanonik Anda, lihat Menemukan ID Akun AWS pengguna kanonik. `Grant`Elemen mengidentifikasi penerima hibah (baik grup Akun AWS atau yang telah ditentukan sebelumnya) dan izin yang diberikan. ACL bawaan ini memiliki satu elemen `Grant` untuk pemilik. Anda memberikan izin dengan menambahkan elemen `Grant`, dengan setiap pemberian yang mengidentifikasi penerimanya dan izinnya.

> **Note**
>
> ACL dapat memiliki hingga 100 pemberian.

Topik

- Siapa itu penerima?
- Izin apa yang dapat saya berikan?
- Nilai aclRequired untuk permintaan Amazon S3 umum
- ACL Sampel
- ACL Terekam

Siapa itu penerima?

Penerima hibah dapat berupa Akun AWS atau salah satu grup Amazon S3 yang telah ditentukan sebelumnya. Anda memberikan izin untuk Akun AWS menggunakan alamat email atau ID pengguna kanonik. Namun, jika Anda memberikan alamat email pada permintaan pemberian Anda, Amazon S3 akan menemukan ID pengguna kanonik untuk akun tersebut dan menambahkannya ke ACL. ACL yang dihasilkan selalu berisi ID pengguna kanonik untuk Akun AWS, bukan alamat email. Akun AWS

Saat Anda memberikan hak akses, Anda menentukan setiap penerima sebagai pasangan `type="value"`, di mana salah satu dari berikut `type` ini:

- `id`— Jika nilai yang ditentukan adalah ID pengguna kanonik dari Akun AWS
- `uri`–jika Anda memberikan izin untuk grup yang telah ditentukan sebelumnya
- `emailAddress`–jika nilai yang disebutkan adalah alamat email dari Akun AWS

> **Important**
>
> Menggunakan alamat email untuk menentukan penerima hanya diberi support di Wilayah AWS berikut ini:
>
> - AS Timur (Virginia Utara)
> - AS Barat (California Utara)
> - AS Barat (Oregon)
> - Asia Pasifik (Singapura)
> - Asia Pasifik (Sydney)
> - Asia Pasifik (Tokyo)
> - Eropa (Irlandia)
> - Amerika Selatan (Sao Paulo)

Untuk daftar semua wilayah dan titik akhir yang didukung Amazon S3, lihat Wilayah dan Titik Akhir di Referensi Umum Amazon Web Services.

Example Contoh: Alamat Email

Misalnya, `x-amz-grant-read` header berikut memberikan izin yang Akun AWS diidentifikasi oleh alamat email untuk membaca data objek dan metadatanya:

```
x-amz-grant-read: emailAddress="xyz@example.com", emailAddress="abc@example.com"
```

Warning

Saat Anda memberikan Akun AWS akses lain ke sumber daya Anda, ketahuilah bahwa mereka Akun AWS dapat mendelegasikan izin mereka kepada pengguna di bawah akun mereka. Ini dikenal sebagai akses lintas akun. Untuk informasi tentang menggunakan akses lintas akun, lihat Membuat Peran untuk Mendelegasikan Izin kepada Pengguna IAM dalam Panduan Pengguna IAM.

Menemukan ID Akun AWS pengguna kanonik

ID pengguna kanonik berkaitan dengan Akun AWS Anda. ID ini adalah string panjang karakter, seperti:

`79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be`

Untuk informasi tentang cara menemukan ID pengguna kanonik untuk akun Anda, lihat Menemukan ID pengguna kanonik untuk Anda Akun AWS di Panduan Referensi Manajemen AWS Akun.

Anda juga dapat mencari ID pengguna kanonik Akun AWS dengan membaca ACL bucket atau objek yang Akun AWS memiliki izin akses. Ketika seseorang Akun AWS diberikan izin oleh permintaan hibah, entri hibah ditambahkan ke ACL dengan ID pengguna kanonik akun.

Note

Jika Anda membuat bucket menjadi publik (tidak disarankan), pengguna yang tidak diautentikasi dapat mengunggah objek ke dalam bucket tersebut. Pengguna

anonim ini tidak memiliki Akun AWS. Ketika pengguna anonim mengunggah objek ke bucket Anda, Amazon S3 akan menambahkan ID pengguna kanonik kustom (`65a011a29cdf8ec533ec3d1ccaae921c`) sebagai pemilik objek dalam ACL. Untuk informasi selengkapnya, lihat bucket Amazon S3 dan kepemilikan objek.

Grup Amazon S3 yang sudah ditentukan sebelumnya

Amazon S3 memiliki serangkaian grup yang telah ditentukan sebelumnya. Saat memberikan akses akun ke sebuah grup, Anda menentukan salah satu URI Amazon S3, bukannya ID pengguna kanonik. Amazon S3 menyediakan grup yang telah ditentukan sebelumnya berikut ini:

- Grup Pengguna yang Diautentikasi–Diwakili oleh `http://acs.amazonaws.com/groups/global/AuthenticatedUsers`.

  Kelompok ini mewakili semua Akun AWS. Izin akses ke grup ini memungkinkan siapa pun Akun AWS untuk mengakses sumber daya. Namun demikian, semua permintaan harus ditandatangani (diautentikasi).

  > **Warning**
  >
  > Saat Anda memberikan akses ke grup Pengguna Terautentikasi, setiap pengguna yang AWS diautentikasi di dunia dapat mengakses sumber daya Anda.

- Grup Semua Pengguna–Diwakili oleh `http://acs.amazonaws.com/groups/global/AllUsers`.

  Izin akses ke grup ini memungkinkan siapa pun di dunia mengakses ke sumber daya. Permintaan dapat ditandatangani (diautentikasi) atau tidak ditandatangani (anonim). Permintaan yang tidak ditandatangani menghilangkan header Autentikasi dalam permintaan.

  > **Warning**
  >
  > Kami sangat merekomendasikan agar Anda tidak pernah memberikan Grup Semua Pengguna izin `WRITE`, `WRITE_ACP`, atau `FULL_CONTROL`. Misalnya, sementara izin `WRITE` tidak mengizinkan non-pemilik untuk menimpa atau menghapus objek yang ada, izin `WRITE` masih memungkinkan siapa pun menyimpan objek dalam bucket Anda, yang akan

ditagihkan kepada Anda. Untuk perincian lebih lanjut tentang izin ini, lihat bagian berikut Izin apa yang dapat saya berikan?.

- Grup Log Pengiriman–Diwakili oleh `http://acs.amazonaws.com/groups/s3/LogDelivery`.

  Izin `WRITE` pada bucket memungkinkan grup ini menulis log akses server (lihat Pencatatan permintaan dengan pencatatan akses server) ke bucket.

> **Note**
>
> Saat menggunakan ACL, penerima hibah dapat berupa satu Akun AWS atau salah satu grup Amazon S3 yang telah ditentukan sebelumnya. Namun, penerima tidak dapat berupa pengguna IAM. Untuk informasi lebih lanjut tentang AWS pengguna dan izin dalam IAM, lihat Menggunakan AWS Identity and Access Management.

Izin apa yang dapat saya berikan?

Tabel berikut mencantumkan serangkaian izin yang didukung Amazon S3 di ACL. Serangkaian izin ACL tersebut sama dengan izin untuk sebuah objek ACL dan bucket ACL. Namun, tergantung pada konteks (ACL bucket atau ACL objek), izin-izin ACL ini memberikan izin untuk bucket atau operasi objek tertentu. Tabel ini mencantumkan izin dan menjelaskan apa artinya dalam konteks objek dan bucket.

Untuk informasi selengkapnya tentang izin ACL di konsol Amazon S3, lihat Mengonfigurasi ACL.

Izin ACL

Izin	Saat diberikan pada bucket	Saat diberikan pada objek
`READ`	Memungkinkan penerima untuk mencantumkan objek di dalam bucket	Memungkinkan penerima untuk membaca data objek dan metadatanya
`WRITE`	Mengizinkan penerima untuk membuat objek baru di dalam bucket. Untuk pemilik bucket dan objek dari objek yang sudah ada, serta mengizinkan penghapusan dan penimpaan objek tersebut.	Tidak berlaku

Izin	Saat diberikan pada bucket	Saat diberikan pada objek
`READ_ACP`	Memungkinkan penerima untuk membaca ACL bucket	Memungkinkan penerima untuk membaca ACL objek
`WRITE_ACP`	Memungkinkan penerima untuk menulis ACL untuk bucket yang berlaku	Memungkinkan penerima untuk menulis ACL untuk objek yang berlaku
`FULL_CONTROL`	Memungkinkan penerima pemberian `READ`, `WRITE`, `READ_ACP`, dan `WRITE_ACP` izin pada bucket	Memungkinkan penerima pemberian `READ`, `WRITE_ACP` , dan `READ_ACP` izin pada objek

> **Warning**
>
> Berhati-hatilah saat memberikan izin akses ke bucket dan S3 Object Anda. Misalnya, memberikan akses `WRITE` ke bucket mengizinkan penerima membuat, menimpa, dan menghapus objek dalam bucket tersebut. Kami sangat menyarankan Anda membaca seluruh Gambaran umum daftar kontrol akses (ACL) bagian sebelum memberikan izin.

Pemetaan izin ACL dan izin kebijakan akses

Sebagaimana yang ditunjukkan dalam tabel sebelumnya, ACL hanya mengizinkan serangkaian izin terbatas, dibandingkan dengan jumlah izin yang dapat Anda tetapkan dalam kebijakan akses (lihat Tindakan kebijakan untuk Amazon S3). Masing-masing izin ini memungkinkan satu operasi Amazon S3 atau lebih.

Tabel berikut menunjukkan cara setiap izin ACL memetakan ke izin kebijakan akses yang sesuai. Seperti yang dapat Anda lihat, kebijakan akses memungkinkan lebih banyak izin dibandingkan ACL. Anda menggunakan ACL utamanya untuk memberikan izin baca/tulis basic, serupa dengan izin sistem file. Untuk informasi lebih lanjut tentang kapan harus menggunakan ACL, lihat Identity and Access Management untuk Amazon S3.

Untuk informasi selengkapnya tentang izin ACL di konsol Amazon S3, lihat Mengonfigurasi ACL.

Izin ACL	Menyesuaikan izin kebijakan akses ketika izin ACL diberikan pada bucket	Menyesuaikan izin kebijakan akses ketika izin ACL diberikan pada objek
READ	s3:ListBucket , s3:ListBucketVersions , dan s3:ListBucketMultipartUploads	s3:GetObject dan s3:GetObjectVersion
WRITE	s3:PutObject Pemilik bucket dapat membuat, menimpa, dan menghapus objek apa pun di bucket, dan pemilik objek memiliki FULL_CONTROL atas objek mereka. Selain itu, ketika penerima merupakan pemilik bucket, memberikan izin WRITE dalam ACL bucket memungkinkan tindakan s3:DeleteObjectVersion yang harus dilakukan pada versi apa pun dalam bucket tersebut.	Tidak berlaku
READ_ACP	s3:GetBucketAcl	s3:GetObjectAcl dan s3:GetObjectVersionAcl
WRITE_ACP	s3:PutBucketAcl	s3:PutObjectAcl dan s3:PutObjectVersionAcl
FULL_CONTROL	Setara dengan pemberian izin READ, WRITE, READ_ACP, dan izin ACL WRITE_ACP . Oleh karena itu, izin ACL ini dipetakan ke kombinasi izin kebijakan akses yang sesuai.	Setara dengan pemberian izin READ, READ_ACP, dan izin ACL WRITE_ACP . Oleh karena itu, izin ACL ini dipetakan ke kombinasi izin kebijakan akses yang sesuai.

Kunci syarat

Saat Anda memberikan izin kebijakan akses, Anda dapat menggunakan kunci kondisi untuk membatasi nilai ACL pada objek dengan menggunakan kebijakan bucket. Tombol konteks berikut sesuai dengan ACL. Anda dapat menggunakan kunci konteks ini untuk mewajibkan penggunaan ACL tertentu dalam permintaan:

- `s3:x-amz-grant-read` - Memerlukan akses baca.
- `s3:x-amz-grant-write` - Memerlukan akses tulis.
- `s3:x-amz-grant-read-acp` - Memerlukan akses baca ke ACL bucket.
- `s3:x-amz-grant-write-acp` - Memerlukan akses tulis ke ACL bucket.
- `s3:x-amz-grant-full-control` - Memerlukan kontrol penuh.
- `s3:x-amz-acl` - Memerlukan ACL Terekam.

Misalnya kebijakan yang melibatkan header spesifik ACL, lihat. Pemberian s3: PutObject izin dengan syarat yang mengharuskan pemilik bucket untuk mendapatkan kontrol penuh Untuk daftar lengkap kunci kondisi khusus Amazon S3, lihat Tindakan, sumber daya, dan kunci kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Nilai **`aclRequired`** untuk permintaan Amazon S3 umum

Untuk mengidentifikasi permintaan Amazon S3 yang memerlukan ACL untuk otorisasi, Anda dapat menggunakan nilai `aclRequired` di log akses server Amazon S3, atau AWS CloudTrail. `aclRequiredNilai` yang muncul di log akses server Amazon S3 CloudTrail atau Amazon S3 bergantung pada operasi mana yang dipanggil dan informasi tertentu tentang pemohon, pemilik objek, dan pemilik bucket. Jika tidak ada ACL yang diperlukan, atau jika Anda menyetel ACL yang `bucket-owner-full-control` dikalengkan, atau jika permintaan diizinkan oleh kebijakan bucket Anda, string `aclRequired` nilainya adalah "`-`" di log akses server Amazon S3 dan tidak ada di dalamnya. CloudTrail

Tabel berikut mencantumkan `aclRequired` nilai yang diharapkan dalam CloudTrail atau log akses server Amazon S3 untuk berbagai operasi API Amazon S3. Anda dapat menggunakan informasi ini untuk memahami operasi Amazon S3 mana yang bergantung pada ACL untuk otorisasi. Dalam tabel berikut, A, B, dan C mewakili akun berbeda yang terkait dengan pemohon, pemilik objek, dan pemilik bucket. Entri dengan tanda bintang (*) menunjukkan salah satu akun A, B, atau C.

> **Note**
>
> Operasi `PutObject` dalam tabel berikut ini, kecuali ditentukan lain, menunjukkan permintaan yang tidak menetapkan ACL, kecuali ACL merupakan ACL `bucket-owner-full-control`. Nilai null untuk `aclRequired` menunjukkan bahwa tidak `aclRequired` ada dalam AWS CloudTrail log.

`aclRequired`nilai untuk CloudTrail

Nama operasi	Peminta	Pemilik objek	Pemilik bucket	Kebijakan bucket memberikan akses	Nilai **`aclRequired`**	Alasan
`GetObject`	A	A	A	Ya atau Tidak	null	Akses akun yang sama
	A	B	A	Ya atau Tidak	null	Akses akun yang sama dengan pemilik bucket diberlakukan
	A	A	B	Ya	null	Akses lintas akun yang diberikan oleh kebijakan bucket
	A	A	B	Tidak	Ya	Akses lintas akun

Nama operasi	Peminta	Pemilik objek	Pemilik bucket	Kebijakan bucket memberikan akses	Nilai **`aclRequired`**	Alasan
						bergantung pada ACL
	A	A	B	Ya	null	Akses lintas akun yang diberikan oleh kebijakan bucket
	A	B	B	Tidak	Ya	Akses lintas akun bergantung pada ACL
	A	B	C	Ya	null	Akses lintas akun yang diberikan oleh kebijakan bucket
	A	B	C	Tidak	Ya	Akses lintas akun bergantung pada ACL
`PutObject`	A	Tidak berlaku	A	Ya atau Tidak	null	Akses akun yang sama

Nama operasi	Peminta	Pemilik objek	Pemilik bucket	Kebijakan bucket memberikan akses	Nilai **`aclRequired`**	Alasan
	A	Tidak berlaku	B	Ya	null	Akses lintas akun yang diberikan oleh kebijakan bucket
	A	Tidak berlaku	B	Tidak	Ya	Akses lintas akun bergantung pada ACL
`PutObject` dengan ACL (kecuali untuk `bucket-owner-full-control`)	*	Tidak berlaku	*	Ya atau Tidak	Ya	Permintaan pemberian ACL
`ListObjects`	A	Tidak berlaku	A	Ya atau Tidak	null	Akses akun yang sama

Nama operasi	Peminta	Pemilik objek	Pemilik bucket	Kebijakan bucket memberikan akses	Nilai **`aclRequired`**	Alasan
	A	Tidak berlaku	B	Ya	null	Akses lintas akun yang diberikan oleh kebijakan bucket
	A	Tidak berlaku	B	Tidak	Ya	Akses lintas akun bergantung pada ACL
`DeleteObject`	A	Tidak berlaku	A	Ya atau Tidak	null	Akses akun yang sama
	A	Tidak berlaku	B	Ya	null	Akses lintas akun yang diberikan oleh kebijakan bucket
	A	Tidak berlaku	B	Tidak	Ya	Akses lintas akun bergantung pada ACL
`PutObjectAcl`	*	*	*	Ya atau Tidak	Ya	Permintaan pemberian ACL

Nama operasi	Peminta	Pemilik objek	Pemilik bucket	Kebijakan bucket memberikan akses	Nilai **aclRequired**	Alasan
`PutBucketAcl`	*	Tidak berlaku	*	Ya atau Tidak	Ya	Permintaan pemberian ACL

> **Note**
>
> Operasi `REST.PUT.OBJECT` dalam tabel berikut ini, kecuali ditentukan lain, menunjukkan permintaan yang tidak menetapkan ACL, kecuali ACL merupakan ACL `bucket-owner-full-control`. String `aclRequired` nilai `"-"` menunjukkan nilai nol di log akses server Amazon S3.

Nilai **`aclRequired`** pencatatan akses server Amazon S3

Nama operasi	Peminta	Pemilik objek	Pemilik bucket	Kebijakan bucket memberikan akses	Nilai **aclRequired**	Alasan
`REST.GET.OBJECT`	A	A	A	Ya atau Tidak	-	Akses akun yang sama
	A	B	A	Ya atau Tidak	-	Akses akun yang sama dengan pemilik bucket diberlakukan

Nama operasi	Peminta	Pemilik objek	Pemilik bucket	Kebijakan bucket memberikan akses	Nilai **`aclRequired`**	Alasan
	A	A	B	Ya	-	Akses lintas akun yang diberikan oleh kebijakan bucket
	A	A	B	Tidak	Ya	Akses lintas akun bergantung pada ACL
	A	B	B	Ya	-	Akses lintas akun yang diberikan oleh kebijakan bucket
	A	B	B	Tidak	Ya	Akses lintas akun bergantung pada ACL

Nama operasi	Peminta	Pemilik objek	Pemilik bucket	Kebijakan bucket memberikan akses	Nilai **`aclRequired`**	Alasan
	A	B	C	Ya	-	Akses lintas akun yang diberikan oleh kebijakan bucket
	A	B	C	Tidak	Ya	Akses lintas akun bergantung pada ACL
`REST.PUT.OBJECT`	A	Tidak berlaku	A	Ya atau Tidak	-	Akses akun yang sama
	A	Tidak berlaku	B	Ya	-	Akses lintas akun yang diberikan oleh kebijakan bucket
	A	Tidak berlaku	B	Tidak	Ya	Akses lintas akun bergantung pada ACL

Nama operasi	Peminta	Pemilik objek	Pemilik bucket	Kebijakan bucket memberikan akses	Nilai **`aclRequired`**	Alasan
`REST.PUT.OBJECT` dengan ACL (kecuali untuk `bucket-owner-full-control`)	*	Tidak berlaku	*	Ya atau Tidak	Ya	Permintaan pemberian ACL
`REST.GET.BUCKET`	A	Tidak berlaku	A	Ya atau Tidak	-	Akses akun yang sama
	A	Tidak berlaku	B	Ya	-	Akses lintas akun yang diberikan oleh kebijakan bucket
	A	Tidak berlaku	B	Tidak	Ya	Akses lintas akun bergantung pada ACL
`REST.DELETE.OBJECT`	A	Tidak berlaku	A	Ya atau Tidak	-	Akses akun yang sama

Nama operasi	Peminta	Pemilik objek	Pemilik bucket	Kebijakan bucket memberikan akses	Nilai **`aclRequired`**	Alasan
	A	Tidak berlaku	B	Ya	-	Akses lintas akun yang diberikan oleh kebijakan bucket
	A	Tidak berlaku	B	Tidak	Ya	Akses lintas akun bergantung pada ACL
`REST.PUT.ACL`	*	*	*	Ya atau Tidak	Ya	Permintaan pemberian ACL

ACL Sampel

ACL sampel pada suatu bucket berikut ini mengidentifikasi pemilik sumber daya dan serangkaian pemberian. Formatnya adalah representasi XML dari ACL dalam Amazon S3 API REST. Pemilik bucket memiliki `FULL_CONTROL` sumber daya. Selain itu, ACL menunjukkan bagaimana izin diberikan pada sumber daya ke dua Akun AWS, diidentifikasi oleh ID pengguna kanonik, dan dua grup Amazon S3 yang telah ditentukan sebelumnya yang dibahas di bagian sebelumnya.

Example

```
<?xml version="1.0" encoding="UTF-8"?>
<AccessControlPolicy xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Owner>
    <ID>Owner-canonical-user-ID</ID>
    <DisplayName>display-name</DisplayName>
  </Owner>
  <AccessControlList>
```

```
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="CanonicalUser">
        <ID>Owner-canonical-user-ID</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>

    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="CanonicalUser">
        <ID>user1-canonical-user-ID</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>WRITE</Permission>
    </Grant>

    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="CanonicalUser">
        <ID>user2-canonical-user-ID</ID>
        <DisplayName>display-name</DisplayName>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>

    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/global/AllUsers</URI>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/s3/LogDelivery</URI>
      </Grantee>
      <Permission>WRITE</Permission>
    </Grant>

  </AccessControlList>
</AccessControlPolicy>
```

ACL Terekam

Amazon S3 mendukung serangkaian pemberian yang telah ditentukan sebelumnya, dikenal sebagai ACLs terekam. Setiap ACL terekam memiliki seperangkat penerima dan izin yang telah ditetapkan. Tabel berikut mencantumkan serangkaian ACL terekam dan pemberian terkait yang telah ditentukan sebelumnya.

ACL Terekam	Berlaku untuk	Izin ditambahkan ke ACL
`private`	Bucket dan objek	Pemilik mendapatkan `FULL_CONTROL` . Tidak ada orang lain yang memiliki hak mengakses (default).
`public-read`	Bucket dan objek	Pemilik mendapatkan `FULL_CONTROL` . Grup `AllUsers` (lihat Siapa itu penerima?) mendapat akses `READ`.
`public-read-write`	Bucket dan objek	Pemilik mendapatkan `FULL_CONTROL` . Grup `AllUsers` mendapatkan akses `READ` dan `WRITE`. memberikan ini pada bucket umumnya tidak disarankan.
`aws-exec-read`	Bucket dan objek	Pemilik mendapatkan `FULL_CONTROL` . Amazon EC2 mendapatkan akses `READ` ke sebuah paketan Amazon Machine Image (AMI) `GET` dari Amazon S3.
`authenticated-read`	Bucket dan objek	Pemilik mendapatkan `FULL_CONTROL` . Grup `AuthenticatedUsers` mendapat akses `READ`.
`bucket-owner-read`	Objek	Pemilik objek mendapat `FULL_CONTROL` . Pemilik bucket mendapat akses `READ`. Jika Anda menyebutk an ACL terekam ini saat membuat bucket, Amazon S3 akan mengabaikannya.
`bucket-owner-full-control`	Objek	Pemilik objek dan pemilik bucket mendapatkan `FULL_CONTROL` atas objek. Jika Anda menyebutk an ACL terekam ini saat membuat bucket, Amazon S3 akan mengabaikannya.

ACL Terekam	Berlaku untuk	Izin ditambahkan ke ACL
`log-delivery-write`	Bucket	Grup `LogDelivery` mendapatkan izin `WRITE` dan `READ_ACP` pada bucket. Untuk informasi lebih lanjut tentang log, lihat (Pencatatan permintaan dengan pencatatan akses server).

> **Note**
>
> Anda hanya dapat menyebutkan salah satu dari ACL terekam ini dalam permintaan Anda.

Anda menentukan ACL terekam dalam permintaan Anda dengan menggunakan header `x-amz-acl` permintaan. Ketika Amazon S3 menerima permintaan dengan ACL terekam dalam permintaan tersebut, itu akan menambahkan pemberian yang telah ditentukan sebelumnya ke ACL dari sumber daya itu.

Mengonfigurasi ACL

Bagian ini menjelaskan cara mengelola izin akses untuk bucket S3 dan objek menggunakan daftar kontrol akses (ACL). Anda dapat menambahkan hibah ke ACL sumber daya Anda menggunakan, (AWS Command Line Interface CLI) AWS Management Console, REST API, atau SDK. AWS

bucket dan izin objek bersifat independen satu sama lain. Objek tidak mewarisi izin dari bucketnya. Misalnya, jika Anda membuat bucket dan memberikan akses tertulis kepada pengguna, Anda tidak dapat mengakses objek pengguna tersebut kecuali jika pengguna memberikan Anda akses secara eksplisit.

Anda dapat memberikan izin kepada Akun AWS pengguna lain atau grup yang telah ditentukan sebelumnya. Pengguna atau grup yang Anda berikan izin disebut penerima pemberian. Secara default, pemilik, Akun AWS yang membuat bucket, memiliki izin penuh.

Setiap izin yang Anda berikan untuk pengguna atau kelompok menambahkan entri ke dalam ACL yang terkait dengan bucket. ACL mencantumkan daftar pemberian, yang mengidentifikasi penerima pemberian dan izin yang diberikan.

Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda, serta menonaktifkan atau mengaktifkan ACL. Secara default, Kepemilikan Objek diatur ke pengaturan yang diberlakukan pemilik Bucket, dan semua ACL dinonaktifkan. Ketika ACL dinonaktifkan, pemilik bucket memiliki semua objek di bucket dan mengelola akses ke objek-objek tersebut secara eksklusif dengan menggunakan kebijakan manajemen akses.

Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL. Kami menyarankan agar ACL dinonaktifkan, kecuali dalam keadaan yang tidak biasa ketika Anda perlu mengontrol akses untuk setiap objek secara individual. Dengan ACL dinonaktifkan, Anda dapat menggunakan kebijakan untuk mengontrol akses ke semua objek di bucket, terlepas dari siapa yang mengunggah objek ke bucket Anda. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

> **Important**
>
> Jika bucket Anda menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek S3, Anda harus menggunakan kebijakan untuk memberikan akses ke bucket Anda, serta objek di dalamnya. Dengan mengaktifkan pengaturan yang diberlakukan pemilik Bucket, permintaan untuk mengatur daftar kontrol akses (ACL) atau memperbarui ACL akan gagal dan mengembalikan kode kesalahan `AccessControlListNotSupported`. Permintaan untuk membaca ACL masih didukung.

> **Warning**
>
> Kami sangat menyarankan agar Anda menghindari pemberian akses tulis ke grup Semua Orang (akses publik) atau grup Pengguna Terautentikasi (semua pengguna yang AWS diautentikasi). Untuk informasi lebih lanjut tentang efek pemberian akses tulis ke grup ini, lihat Grup Amazon S3 yang sudah ditentukan sebelumnya.

Menggunakan konsol S3 untuk mengatur izin ACL untuk bucket

Konsol tersebut menampilkan pemberian akses gabungan untuk penerima pemberian duplikat. Untuk melihat daftar lengkap ACL, gunakan Amazon S3 REST API AWS CLI, AWS atau SDK.

Tabel berikut menunjukkan izin ACL yang dapat Anda konfigurasikan untuk bucket di konsol Amazon S3.

Izin ACL konsol Amazon S3 untuk bucket

Izin konsol	Izin ACL	Akses
Objek-Daftar	`READ`	Memungkinkan penerima untuk mencantumkan objek di dalam bucket.
Objek-Tulis	`WRITE`	Mengizinkan penerima untuk membuat objek baru di dalam bucket. Untuk pemilik bucket dan objek dari objek yang sudah ada, serta mengizinkan penghapusan dan penimpaan objek tersebut.
Bucket ACL-Baca	`READ_ACP`	Memungkinkan penerima untuk membaca ACL bucket.
Bucket ACL-Tulis	`WRITE_ACP`	Memungkinkan penerima untuk menulis ACL untuk bucket yang berlaku.
Semua orang (akses publik): Objek-Daftar	`READ`	Hibah akses baca publik untuk objek dalam bucket. Ketika Anda memberikan akses daftar ke Semua orang (akses publik), siapa pun di dunia ini dapat mengakses objek yang ada dalam bucket.
Semua orang (akses publik): Bucket ACL-Baca	`READ_ACP`	Hibah akses baca publik untuk objek dalam bucket ACL. Ketika Anda memberikan akses baca ke Semua orang (akses publik), siapa pun di dunia ini dapat mengakses objek yang ada dalam bucket ACL.

Untuk informasi selengkapnya tentang izin ACL, lihat Gambaran umum daftar kontrol akses (ACL).

> **Important**
>
> Jika bucket Anda menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek S3, Anda harus menggunakan kebijakan untuk memberikan akses ke bucket Anda, serta objek di dalamnya. Dengan mengaktifkan pengaturan yang diberlakukan pemilik Bucket, permintaan untuk mengatur daftar kontrol akses

(ACL) atau memperbarui ACL akan gagal dan mengembalikan kode kesalahan `AccessControlListNotSupported`. Permintaan untuk membaca ACL masih didukung.

Untuk mengatur izin ACL untuk bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di Bucket Anda, pilih nama bucket yang ingin Anda tetapkan izinnya.
3. Pilih Izin.
4. Di bawah Daftar kontrol akses, memilih Edit.

   Anda dapat mengedit izin ACL berikut untuk bucket:

   Objek

   - Daftar–Mengizinkan penerima untuk mencantumkan objek di dalam bucket.
   - Tulis–Mengizinkan penerima untuk membuat objek baru di dalam bucket. Untuk pemilik bucket dan objek dari objek yang sudah ada, serta mengizinkan penghapusan dan penimpaan objek tersebut.

     Di konsol S3, Anda hanya dapat memberikan akses tulis ke grup pengiriman log S3 dan pemilik bucket (milik Anda Akun AWS). Kami sangat merekomendasikan Anda untuk tidak memberikan akses tulis untuk penerima pemberian lainnya. Namun, jika Anda perlu memberikan akses tulis, Anda dapat menggunakan AWS CLI, AWS SDK, atau REST API.

   ACL Bucket

   - Baca–Memungkinkan penerima untuk membaca ACL bucket.
   - Tulis–Memungkinkan penerima untuk menulis ACL untuk bucket yang berlaku.
5. Untuk mengubah izin pemilik bucket, di samping pemilik Bucket (Anda Akun AWS), hapus atau pilih dari izin ACL berikut:

   - Objek–Daftar atau Tulis
   - ACL bucket–Baca atau Tulis

Pemilik mengacu pada Pengguna root akun AWS, bukan pengguna AWS Identity and Access Management IAM. Untuk informasi selengkapnya tentang pengguna root, lihat Pengguna root akun AWS dalam Panduan Pengguna IAM.

6. Untuk memberikan atau membatalkan izin untuk masyarakat umum (semua orang di internet), di samping Semua orang (akses publik), hapus atau pilih dari izin ACL berikut:

   - Objek–Daftar
   - ACL bucket–Baca

   > **Warning**
   >
   > Berhati-hatilah saat memberikan Semua akses publik kelompok ke bucket S3 Anda. Ketika Anda memberikan akses ke grup ini, siapa pun di dunia dapat mengakses bucket Anda. Kami sangat menyarankan Anda untuk tidak pernah memberikan akses tulis kepada publik ke bucket S3 Anda.

7. Untuk memberikan atau membatalkan izin bagi siapa pun yang memiliki Akun AWS, di samping grup Pengguna Terautentikasi (siapa pun yang memiliki Akun AWS), hapus atau pilih dari izin ACL berikut:

   - Objek–Daftar
   - ACL bucket–Baca

8. Untuk memberikan atau membatalkan izin bagi Amazon S3 untuk tulis log akses server ke bucket, di bawah Grup pengiriman log S3, hapus atau pilih dari izin ACL berikut:

   - Objek–Daftar atau Tulis
   - ACL bucket–Baca atau Tulis

   Jika bucket disiapkan sebagai bucket target untuk menerima catatan akses, izin bucket harus mengizinkan Log Pengiriman kelompok menulis akses ke bucket. Saat Anda mengaktifkan log masuk akses server ke dalam bucket, konsol Amazon S3 memberikan akses tertulis ke Log Pengiriman kelompok untuk bucket target yang Anda pilih untuk menerima log. Untuk informasi selengkapnya tentang pencatatan log akses server, lihat Mengaktifkan pencatatan akses server Amazon S3.

9. Untuk memberikan akses ke yang lain Akun AWS, lakukan hal berikut:

a. Pilih Tambahkan penerima.
b. Di kotak Penerima, memasukkan ID kanonik Akun AWS yang lain.
c. Pilih dari izin ACL berikut:
   - Objek–Daftar atau Tulis
   - ACL bucket–Baca atau Tulis

> **Warning**
> Saat Anda memberikan Akun AWS akses lain ke sumber daya Anda, ketahuilah bahwa mereka Akun AWS dapat mendelegasikan izin mereka kepada pengguna di bawah akun mereka. Ini dikenal sebagai akses lintas akun. Untuk informasi tentang menggunakan akses lintas akun, lihat Membuat Peran untuk Mendelegasikan Izin kepada Pengguna IAM dalam Panduan Pengguna IAM.

10. Untuk menghapus akses ke yang lain Akun AWS, di bawah Access for other Akun AWS, pilih Hapus.
11. Untuk menyimpan perubahan Anda, memilih Simpan perubahan.

Menggunakan konsol S3 untuk mengatur izin ACL untuk objek

Konsol tersebut menampilkan pemberian akses gabungan untuk penerima pemberian duplikat. Untuk melihat daftar lengkap ACL, gunakan Amazon S3 REST API AWS CLI, AWS atau SDK. Tabel berikut menunjukkan izin ACL yang dapat Anda konfigurasikan untuk bucket di konsol Amazon S3.

Izin ACL konsol Amazon S3 untuk objek

Izin konsol	Izin ACL	Akses
Objek-Baca	`READ`	Memungkinkan penerima untuk membaca data objek dan metadatanya.
Objek ACL-Baca	`READ_ACP`	Mengizinkan penerima untuk membaca ACL objek.
Objek ACL-Tulis	`WRITE_ACP`	Memungkinkan penerima untuk menulis ACL untuk objek yang berlaku

Untuk informasi selengkapnya tentang izin ACL, lihat Gambaran umum daftar kontrol akses (ACL).

> **Important**
>
> Jika bucket Anda menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek S3, Anda harus menggunakan kebijakan untuk memberikan akses ke bucket Anda, serta objek di dalamnya. Dengan mengaktifkan pengaturan yang diberlakukan pemilik Bucket, permintaan untuk mengatur daftar kontrol akses (ACL) atau memperbarui ACL akan gagal dan mengembalikan kode kesalahan `AccessControlListNotSupported`. Permintaan untuk membaca ACL masih didukung.

Untuk mengatur izin ACL untuk objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Bucket, pilih nama bucket yang berisi objek.
3. Di daftar Objek, memilih nama objek yang ingin Anda atur izinnya.
4. Pilih Izin.
5. Di bagian bawah Daftar kontrol akses (ACL), memilih Edit.

   Anda dapat mengedit izin ACL berikut untuk objek:

   Objek

   - Baca–Memungkinkan penerima untuk membaca data objek dan metadatanya.

   ACL Objek

   - Baca–Mengizinkan penerima untuk baca ACL objek.
   - Tulis–Mengizinkan penerima untuk tulis ACL untuk objek yang dapat diaplikasikan. Di konsol S3, Anda hanya dapat memberikan akses tulis ke pemilik bucket (milik Anda Akun AWS). Kami sangat merekomendasikan Anda untuk tidak memberikan akses tulis untuk penerima pemberian lainnya. Namun, jika Anda perlu memberikan akses tulis, Anda dapat menggunakan AWS CLI, AWS SDK, atau REST API.
6. Anda dapat mengelola izin akses objek untuk hal berikut:

a. Akses untuk pemilik objek

Pemilik mengacu pada Pengguna root akun AWS, dan bukan pengguna AWS Identity and Access Management IAM. Untuk informasi selengkapnya tentang pengguna root, lihat Pengguna root akun AWS dalam Panduan Pengguna IAM.

Untuk mengubah izin akses objek pemilik, di bawah Akses untuk pemilik objek, pilih AWS Akun Anda (pemilik).

Pilih kotak periksa untuk izin yang ingin Anda berikan, lalu memilih Simpan.

b. Akses untuk lainnya Akun AWS

Untuk memberikan izin kepada AWS pengguna dari yang lain Akun AWS, di bawah Access for other Akun AWS, pilih Tambah akun. Di bidang Masukkan ID, masukkan ID kanonik AWS pengguna yang ingin Anda berikan izin objek. Untuk informasi tentang menemukan ID kanonik, lihat Akun AWS Pengidentifikasi Anda di. Referensi Umum Amazon Web Services Anda dapat menambahkan sebanyak 99 pengguna.

Pilih kotak periksa untuk izin yang ingin Anda berikan ke pengguna, lalu memilih Simpan. Untuk menampilkan informasi tentang izin, memilih ikon Bantuan.

c. Akses publik

Untuk memberikan akses ke objek Anda ke masyarakat umum (semua orang di dunia), di bawah Akses publik, memilih Semua orang. Pemberian izin akses publik berarti siapa pun di dunia dapat mengakses objek tersebut.

Pilih kotak centang untuk izin yang ingin Anda berikan, lalu pilih Simpan.

> **Warning**
>
> - Berhati-hatilah saat memberikan Semua akses grup anonim ke objek Amazon S3. Ketika Anda memberikan akses ke grup ini, siapa pun di dunia ini dapat mengakses objek Anda. Jika Anda perlu memberikan akses ke semua orang, kami sangat menyarankan agar Anda hanya memberikan izin untuk Baca objek.
> - Kami sangat menyarankan Anda untuk tidak memberikan Semua kelompok menulis izin objek. Dengan begitu, siapa pun dapat menimpa izin ACL untuk objek tersebut.

Menggunakan AWS SDK

Bagian ini menyediakan contoh-contoh cara mengonfigurasikan pemberian daftar kontrol akses (ACL) pada bucket dan objek.

> **Important**
>
> Jika bucket Anda menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek S3, Anda harus menggunakan kebijakan untuk memberikan akses ke bucket Anda, serta objek di dalamnya. Dengan mengaktifkan pengaturan yang diberlakukan pemilik Bucket, permintaan untuk mengatur daftar kontrol akses (ACL) atau memperbarui ACL akan gagal dan mengembalikan kode kesalahan `AccessControlListNotSupported`. Permintaan untuk membaca ACL masih didukung.

Java

Bagian ini menyediakan contoh-contoh cara mengonfigurasikan pemberian daftar kontrol akses (ACL) pada bucket dan objek. Contoh pertama membuat bucket dengan ACL terekam (lihat ACL Terekam), membuat daftar pemberian izin kustom, lalu mengganti ACL terekam dengan ACL yang berisi pemberian kustom. Contoh kedua menunjukkan cara memodifikasi ACL dengan menggunakan metode `AccessControlList.grantPermission()`.

Example Membuat bucket dan menentukan ACL terekam yang memberikan izin ke grup pengiriman log S3

Contoh ini membuat bucket. Dalam permintaan tersebut, contoh menyebutkan ACL terekam yang memberikan izin kepada grup Pengiriman Log untuk menulis log ke bucket.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.IOException;
import java.util.ArrayList;

public class CreateBucketWithACL {
```

```
   public static void main(String[] args) throws IOException {
       Regions clientRegion = Regions.DEFAULT_REGION;
       String bucketName = "*** Bucket name ***";
       String userEmailForReadPermission = "*** user@example.com ***";

       try {
           AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                   .withRegion(clientRegion)
                   .build();

           // Create a bucket with a canned ACL. This ACL will be replaced by the
           // setBucketAcl()
           // calls below. It is included here for demonstration purposes.
           CreateBucketRequest createBucketRequest = new
CreateBucketRequest(bucketName, clientRegion.getName())
                   .withCannedAcl(CannedAccessControlList.LogDeliveryWrite);
           s3Client.createBucket(createBucketRequest);

           // Create a collection of grants to add to the bucket.
           ArrayList<Grant> grantCollection = new ArrayList<Grant>();

           // Grant the account owner full control.
           Grant grant1 = new Grant(new
CanonicalGrantee(s3Client.getS3AccountOwner().getId()),
                   Permission.FullControl);
           grantCollection.add(grant1);

           // Grant the LogDelivery group permission to write to the bucket.
           Grant grant2 = new Grant(GroupGrantee.LogDelivery, Permission.Write);
           grantCollection.add(grant2);

           // Save grants by replacing all current ACL grants with the two we just
created.
           AccessControlList bucketAcl = new AccessControlList();
           bucketAcl.grantAllPermissions(grantCollection.toArray(new Grant[0]));
           s3Client.setBucketAcl(bucketName, bucketAcl);

           // Retrieve the bucket's ACL, add another grant, and then save the new
ACL.
           AccessControlList newBucketAcl = s3Client.getBucketAcl(bucketName);
           Grant grant3 = new Grant(new
EmailAddressGrantee(userEmailForReadPermission), Permission.Read);
           newBucketAcl.grantAllPermissions(grant3);
```

```
            s3Client.setBucketAcl(bucketName, newBucketAcl);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example Memperbarui ACL pada objek yang sudah ada

Contoh ini memperbarui ACL pada objek. Contoh tersebut melakukan tugas sebagai berikut:

- Mengambil ACL objek
- Menghapus ACL dengan menghapus semua izin yang ada
- Menambahkan dua izin: akses penuh kepada pemilik, dan WRITE_ACP (lihat Izin apa yang dapat saya berikan?) kepada pengguna yang diidentifikasi melalui alamat email
- Menyimpan ACL ke objek

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.AccessControlList;
import com.amazonaws.services.s3.model.CanonicalGrantee;
import com.amazonaws.services.s3.model.EmailAddressGrantee;
import com.amazonaws.services.s3.model.Permission;

import java.io.IOException;

public class ModifyACLExistingObject {

    public static void main(String[] args) throws IOException {
```

```
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Key name ***";
        String emailGrantee = "*** user@example.com ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Get the existing object ACL that we want to modify.
            AccessControlList acl = s3Client.getObjectAcl(bucketName, keyName);

            // Clear the existing list of grants.
            acl.getGrantsAsList().clear();

            // Grant a sample set of permissions, using the existing ACL owner for
 Full
            // Control permissions.
            acl.grantPermission(new CanonicalGrantee(acl.getOwner().getId()),
 Permission.FullControl);
            acl.grantPermission(new EmailAddressGrantee(emailGrantee),
 Permission.WriteAcp);

            // Save the modified ACL back to the object.
            s3Client.setObjectAcl(bucketName, keyName, acl);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Example Membuat bucket dan menentukan ACL terekam yang memberikan izin ke grup pengiriman log S3

Contoh C# ini membuat bucket. Dalam permintaan tersebut, kode juga menentukan ACL terekam yang memberikan izin kepada grup Pengiriman Log untuk menulis log ke dalam bucket.

Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class ManagingBucketACLTest
    {
        private const string newBucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            CreateBucketUseCannedACLAsync().Wait();
        }

        private static async Task CreateBucketUseCannedACLAsync()
        {
            try
            {
                // Add bucket (specify canned ACL).
                PutBucketRequest putBucketRequest = new PutBucketRequest()
                {
                    BucketName = newBucketName,
                    BucketRegion = S3Region.EUW1, // S3Region.US,
                                                  // Add canned ACL.
```

```
                    CannedACL = S3CannedACL.LogDeliveryWrite
                };
                PutBucketResponse putBucketResponse = await
 client.PutBucketAsync(putBucketRequest);

                // Retrieve bucket ACL.
                GetACLResponse getACLResponse = await client.GetACLAsync(new
 GetACLRequest
                {
                    BucketName = newBucketName
                });
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine("S3 error occurred. Exception: " +
 amazonS3Exception.ToString());
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception: " + e.ToString());
            }
        }
    }
}
```

Example Memperbarui ACL pada objek yang sudah ada

Contoh C# ini memperbarui ACL pada objek yang sudah ada. Contoh tersebut melakukan tugas sebagai berikut:

- Mengambil ACL objek.
- Menghapus ACL dengan menghapus semua izin yang ada.
- Menambahkan dua izin: akses penuh kepada pemilik, dan WRITE_ACP kepada pengguna yang diidentifikasi melalui alamat email.
- Menyimpan ACL dengan mengirim permintaan `PutAcl`.

Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
```

```
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class ManagingObjectACLTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** object key name ***";
        private const string emailAddress = "*** email address ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;
        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            TestObjectACLTestAsync().Wait();
        }
        private static async Task TestObjectACLTestAsync()
        {
            try
            {
                    // Retrieve the ACL for the object.
                    GetACLResponse aclResponse = await client.GetACLAsync(new
 GetACLRequest
                    {
                        BucketName = bucketName,
                        Key = keyName
                    });

                    S3AccessControlList acl = aclResponse.AccessControlList;

                    // Retrieve the owner (we use this to re-add permissions after
 we clear the ACL).
                    Owner owner = acl.Owner;

                    // Clear existing grants.
                    acl.Grants.Clear();

                    // Add a grant to reset the owner's full permission (the
 previous clear statement removed all permissions).
```

```
                    S3Grant fullControlGrant = new S3Grant
                    {
                        Grantee = new S3Grantee { CanonicalUser = owner.Id },
                        Permission = S3Permission.FULL_CONTROL

                    };

                    // Describe the grant for the permission using an email address.
                    S3Grant grantUsingEmail = new S3Grant
                    {
                        Grantee = new S3Grantee { EmailAddress = emailAddress },
                        Permission = S3Permission.WRITE_ACP
                    };
                    acl.Grants.AddRange(new List<S3Grant> { fullControlGrant,
 grantUsingEmail });

                    // Set a new ACL.
                    PutACLResponse response = await client.PutACLAsync(new
 PutACLRequest
                    {
                        BucketName = bucketName,
                        Key = keyName,
                        AccessControlList = acl
                    });
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine("An AmazonS3Exception was thrown. Exception: " +
 amazonS3Exception.ToString());
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception: " + e.ToString());
            }
        }
    }
}
```

Penggunaan API REST

Amazon S3 API mengaktifkan Anda untuk mengatur ACL saat Anda membuat bucket atau objek. Amazon S3 juga menyediakan API untuk menetapkan ACL pada bucket atau objek yang ada. API ini memberikan metode-metode berikut untuk mengatur ACL:

- Tetapkan ACL menggunakan header permintaan— Ketika Anda mengirim permintaan untuk membuat sumber daya (bucket atau objek), Anda mengatur ACL dengan menggunakan header permintaan. Dengan menggunakan header ini, Anda dapat menentukan ACL terekam atau menentukan pemberian secara jelas (mengidentifikasi penerima dan izin secara jelas).
- Tetapkan ACL menggunakan isi permintaan— Ketika Anda mengirim permintaan untuk menetapkan ACL pada sumber daya yang sudah ada, Anda dapat mengatur ACL baik pada header permintaan atau isinya.

Untuk informasi tentang support API REST untuk mengelola ACL, lihat bagian berikut dalam Referensi API Amazon Simple Storage Service:

- GET Bucket acl
- PUT Bucket acl
- GET Object acl
- PUT Objek acl
- PUT Objek
- PUT Bucket
- PUT Objek-Salin
- Mulai Unggahan Multibagian

> **Important**
>
> Jika bucket Anda menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek S3, Anda harus menggunakan kebijakan untuk memberikan akses ke bucket Anda, serta objek di dalamnya. Dengan mengaktifkan pengaturan yang diberlakukan pemilik Bucket, permintaan untuk mengatur daftar kontrol akses (ACL) atau memperbarui ACL akan gagal dan mengembalikan kode kesalahan `AccessControlListNotSupported`. Permintaan untuk membaca ACL masih didukung.

Daftar Kontrol Akses (ACL)-Header Permintaan Spesifik

Anda dapat menggunakan header untuk memberikan izin berbasis daftar kontrol akses (ACL). Secara bawaan, semua objek bersifat privat. Hanya pemilik yang memiliki kontrol akses penuh. Saat menambahkan objek baru, Anda dapat memberikan izin ke individu Akun AWS atau grup yang telah ditentukan sebelumnya yang ditentukan oleh Amazon S3. Izin ini kemudian ditambahkan ke Daftar Kontrol Akses (ACL) pada objek. Untuk informasi selengkapnya, lihat Gambaran umum daftar kontrol akses (ACL).

Dengan operasi ini, Anda dapat memberikan izin akses dengan menggunakan salah satu dari dua metode berikut:

- ACL Terekam (**`x-amz-acl`**)—Amazon S3 mendukung serangkaian pemberian yang telah ditentukan sebelumnya, dikenal sebagai ACLs terekam. Setiap ACL terekam memiliki seperangkat penerima dan izin yang telah ditetapkan. Untuk informasi selengkapnya, lihat ACL Terekam.
- Izin Akses — Untuk secara eksplisit memberikan izin akses ke grup Akun AWS atau tertentu, gunakan header berikut. Setiap header memetakan ke izin tertentu yang didukung Amazon S3 di ACL. Untuk informasi selengkapnya, lihat Gambaran umum daftar kontrol akses (ACL). Pada header tersebut, Anda menentukan daftar penerima yang mendapatkan izin kustom.
  - x-amz-grant-read
  - x-amz-grant-write
  - x-amz-grant-read-ACP
  - x-amz-grant-write-ACP
  - x-amz-grant-full-kontrol

Menggunakan AWS CLI

Untuk informasi selengkapnya tentang mengelola ACL menggunakan AWS CLI, lihat put-bucket-acldi AWS CLI Command Reference.

> **Important**
>
> Jika bucket Anda menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek S3, Anda harus menggunakan kebijakan untuk memberikan akses ke bucket Anda, serta objek di dalamnya. Dengan mengaktifkan pengaturan yang diberlakukan pemilik Bucket, permintaan untuk mengatur daftar kontrol akses

(ACL) atau memperbarui ACL akan gagal dan mengembalikan kode kesalahan `AccessControlListNotSupported`. Permintaan untuk membaca ACL masih didukung.

Contoh kebijakan untuk ACL

Anda dapat menggunakan kunci kondisi dalam kebijakan bucket untuk mengontrol akses ke Amazon S3.

Topik

- Pemberian s3: PutObject izin dengan syarat yang mengharuskan pemilik bucket untuk mendapatkan kontrol penuh
- Pemberian s3: PutObject izin dengan kondisi di header x-amz-acl

Pemberian s3: PutObject izin dengan syarat yang mengharuskan pemilik bucket untuk mendapatkan kontrol penuh

Operasi PUT Objek memungkinkan header spesifik daftar kontrol akses (ACL) yang dapat Anda gunakan untuk memberikan izin berbasis ACL. Dengan menggunakan kunci ini, pemilik bucket dapat menetapkan kondisi-kondisi untuk mengharuskan izin akses kustom ketika pengguna mengunggah objek.

Seandainya ketika Akun A memiliki bucket, dan administrator akun ingin memberikan izin kepada Dave, pengguna di Akun B, untuk mengunggah objek. Secara bawaan, objek yang diunggah Dave dimiliki oleh Akun B, dan Akun A tidak memiliki izin pada objek tersebut. Karena pemilik bucket membayar tagihan, pemilik bucket menginginkan izin penuh pada objek yang diunggah oleh Dave. Administrator Akun A dapat melakukan hal ini dengan memberikan izin `s3:PutObject` kepada Dave, dengan kondisi bahwa permintaan tersebut mencakup header spesifik ACL yang memberikan izin penuh secara jelas atau menggunakan ACL terekam. Untuk informasi lebih lanjut, lihat PUT Objek.

Membutuhkan x-amz-full-control header

Anda dapat mengharuskan header `x-amz-full-control` pada permintaan dengan izin kontrol penuh kepada pemilik bucket. Kebijakan bucket berikut ini memberikan izin `s3:PutObject` bagi pengguna Dave dengan syarat yang menggunakan kunci syarat `s3:x-amz-grant-full-control`, yang mengharuskan permintaan untuk menyertakan header `x-amz-full-control`.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "statement1",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::AccountB-ID:user/Dave"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::awsexamplebucket1/*",
      "Condition": {
        "StringEquals": {
          "s3:x-amz-grant-full-control": "id=AccountA-CanonicalUserID"
        }
      }
    }
  ]
}
```

> **Note**
>
> Contoh ini adalah tentang izin lintas akun. Namun, jika Dave (yang mendapatkan izin) milik pemilik Akun AWS ember, izin bersyarat ini tidak diperlukan. Hal ini karena akun induk yang merupakan pemilik Dave memiliki objek yang diunggah pengguna.

Menambahkan penolakan eksplisit

Kebijakan bucket sebelumnya memberikan izin bersyarat kepada pengguna Dave yang ada di Akun B. Meskipun kebijakan ini berlaku, Dave mungkin mendapatkan izin yang sama tanpa persyaratan apa pun melalui kebijakan lain. Misalnya, Dave dapat menjadi bagian dari sebuah kelompok, dan Anda memberikan kepada kelompok tersebut izin `s3:PutObject` tanpa syarat apa pun. Untuk menghindari celah izin seperti itu, Anda dapat membuat kebijakan akses yang lebih ketat dengan menambahkan penolakan secara eksplisit. Dalam contoh ini, Anda secara jelas menolak izin pengunggahan pengguna Dave jika dia tidak menyertakan header yang diperlukan dalam permintaan yang memberikan izin penuh kepada pemilik bucket. Penolakan eksplisit selalu menggantikan izin lain yang diberikan. Berikut ini adalah contoh kebijakan akses yang direvisi dengan ditambahkan penolakan eksplisit.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "statement1",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::AccountB-ID:user/AccountBadmin"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::awsexamplebucket1/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-grant-full-control": "id=AccountA-CanonicalUserID"
                }
            }
        },
        {
            "Sid": "statement2",
            "Effect": "Deny",
            "Principal": {
                "AWS": "arn:aws:iam::AccountB-ID:user/AccountBadmin"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::awsexamplebucket1/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-grant-full-control": "id=AccountA-CanonicalUserID"
                }
            }
        }
    ]
}
```

Uji kebijakan dengan AWS CLI

Jika Anda memiliki dua Akun AWS, Anda dapat menguji kebijakan menggunakan AWS Command Line Interface (AWS CLI). Anda melampirkan kebijakan dan menggunakan kredensyal Dave untuk menguji izin menggunakan perintah berikut. AWS CLI `put-object` Anda memberikan kredensial Dave dengan menambahkan parameter `--profile`. Anda memberikan izin kontrol penuh kepada pemilik bucket dengan menambahkan parameter `--grant-full-control`. Untuk informasi selengkapnya tentang pengaturan dan penggunaan AWS CLI, lihat[Mengembangkan dengan Amazon S3 menggunakan AWS CLI].

```
aws s3api put-object --bucket examplebucket --key HappyFace.jpg --body c:\HappyFace.jpg
 --grant-full-control id="AccountA-CanonicalUserID" --profile AccountBUserProfile
```

Membutuhkan x-amz-acl header

Anda dapat mengharuskan header `x-amz-acl` dengan ACL terekam yang memberikan izin kontrol penuh kepada pemilik bucket. Untuk mengharuskan header `x-amz-acl` pada permintaan, Anda dapat mengganti pasangan nilai kunci dalam pemblokiran `Condition` dan tentukan kunci syarat `s3:x-amz-acl`, sebagaimana yang ditunjukkan pada contoh berikut.

```
"Condition": {
    "StringEquals": {
        "s3:x-amz-acl": "bucket-owner-full-control"
    }
}
```

Untuk menguji izin menggunakan AWS CLI, Anda menentukan `--acl` parameter. AWS CLI Kemudian menambahkan `x-amz-acl` header ketika mengirim permintaan.

```
aws s3api put-object --bucket examplebucket --key HappyFace.jpg --body c:\HappyFace.jpg
 --acl "bucket-owner-full-control" --profile AccountBadmin
```

Pemberian s3: PutObject izin dengan kondisi di header x-amz-acl

Kebijakan bucket berikut memberikan `s3:PutObject` izin untuk dua orang Akun AWS jika permintaan menyertakan `x-amz-acl` header yang membuat objek dapat dibaca secara publik. Pemblokiran `Condition` menggunakan kondisi `StringEquals` dan diberi pasangan nilai kunci, `"s3:x-amz-acl":["public-read"]`, untuk evaluasi. Dalam pasangan nilai kunci, `s3:x-amz-acl` adalah kunci spesifik–Amazon S3, sebagaimana yang ditunjukkan dengan prefiks `s3:`.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
         "Sid":"AddCannedAcl",
            "Effect":"Allow",
         "Principal": {
                "AWS": [
                "arn:aws:iam::Account1-ID:root",
    "arn:aws:iam::Account2-ID:root"
    ]
```

```
            },
         "Action":"s3:PutObject",
            "Resource": ["arn:aws:s3:::awsexamplebucket1/*"],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl":["public-read"]
                }
            }
        }
    ]
}
```

> **Important**
>
> Tidak semua kondisi masuk akal untuk semua tindakan. Misalnya, memasukkan kondisi `s3:LocationConstraint` dalam kebijakan yang memberikan izin `s3:CreateBucket` Amazon S3 adalah hal yang masuk akal. Namun, tidak masuk akal untuk memasukkan syarat ini pada kebijakan yang memberikan izin `s3:GetObject`. Amazon S3 dapat menguji jenis kesalahan semantik seperti ini yang melibatkan syarat spesifik Amazon S3. Namun, jika Anda membuat kebijakan untuk pengguna IAM dan Anda menyertakan kondisi Amazon S3 yang tidak valid secara semantis, tidak akan ada kesalahan yang dilaporkan karena IAM tidak dapat memvalidasi syarat Amazon S3.

Melakukan blok akses publik ke penyimpanan Amazon S3 Anda

Fitur Blokir Akses Publik Amazon S3 menyediakan pengaturan untuk titik akses, bucket, dan akun untuk membantu Anda mengelola akses publik ke sumber daya Amazon S3. Secara bawaan, bucket baru, titik akses, dan objek baru tidak mengizinkan akses publik. Namun, pengguna dapat memodifikasi kebijakan bucket, kebijakan titik akses, atau izin objek untuk memungkinkan akses publik. Pengaturan S3 Blokir Akses Publik menolak kebijakan-kebijakan dan izin tersebut sehingga Anda dapat membatasi akses publik ke sumber daya.

Dengan S3 Blokir Akses Publik, administrator akun dan pemilik bucket dapat dengan mudah menyiapkan kontrol terpusat untuk membatasi akses publik ke sumber daya Amazon S3 mereka yang diberlakukan terlepas dari bagaimana sumber daya tersebut dibuat.

Untuk petunjuk tentang mengonfigurasi akses blok publik, lihat Mengonfigurasi blokir akses publik.

Saat Amazon S3 menerima permintaan untuk mengakses bucket atau objek, itu akan menentukan apakah akun dari bucket atau pemilik bucket menerapkan blokir akses publik. Jika permintaan tersebut dibuat melalui titik akses, maka Amazon S3 juga memeriksa pengaturan blokir akses publik untuk titik akses tersebut. Jika ada pengaturan blokir akses publik yang sudah ada yang melarang akses yang diminta tersebut, maka Amazon S3 akan menolak permintaan tersebut.

Blokir Akses Publik Amazon S3 menyediakan empat pengaturan. Pengaturan ini bersifat independen dan dapat digunakan dalam kombinasi apa pun. Setiap pengaturan dapat diterapkan ke titik akses, bucket, atau seluruh Akun AWS. Jika pengaturan blokir akses publik untuk titik akses, bucket, atau akun berbeda, maka Amazon S3 akan menerapkan kombinasi yang paling ketat dari pengaturan titik akses, bucket, dan akun.

Ketika Amazon S3 mengevaluasi apakah suatu operasi dilarang oleh pengaturan blokir akses publik, itu akan menolak permintaan apa pun yang melanggar titik akses, bucket, atau pengaturan akun.

> **Important**
>
> Akses publik diberikan kepada bucket dan objek melalui daftar kontrol akses (ACL), kebijakan titik akses, kebijakan bucket, atau semuanya. Untuk membantu memastikan bahwa semua titik akses Amazon S3, bucket, dan objek Anda diblokir akses publiknya, kami menyarankan Anda untuk mengaktifkan keempat pengaturan guna memblokir akses publik untuk akun Anda. Pengaturan ini memblokir akses publik untuk semua bucket dan titik akses saat ini dan mendatang.
>
> Sebelum menerapkan pengaturan ini, verifikasi bahwa aplikasi Anda akan bekerja dengan baik tanpa akses publik. Jika Anda memerlukan beberapa tingkat akses publik ke bucket atau objek Anda—misalnya, untuk menjadi host situs web statis seperti yang dijelaskan di Hosting situs web statis menggunakan Amazon S3—Anda dapat menyesuaikan pengaturan individual agar sesuai dengan kasus penggunaan penyimpanan Anda.
>
> Mengaktifkan Blokir Akses Publik membantu melindungi sumber daya Anda dengan mencegah akses publik diberikan melalui kebijakan sumber daya atau daftar kontrol akses (ACL) yang langsung dilampirkan ke sumber daya S3. Selain mengaktifkan Blokir Akses Publik, periksa kebijakan berikut dengan cermat untuk mengonfirmasi bahwa kebijakan tersebut tidak memberikan akses publik:
>
> - Kebijakan berbasis identitas yang dilampirkan pada AWS prinsipal terkait (misalnya, peran IAM)

- Kebijakan berbasis sumber daya yang dilampirkan pada AWS sumber daya terkait (misalnya, kunci (KMS AWS Key Management Service))

> **Note**
>
> - Anda dapat mengaktifkan pengaturan blokir akses publik hanya untuk titik akses, bucket, dan Akun AWS. Amazon S3 tidak mendukung pengaturan blokir akses publik secara per-objek.
> - Saat Anda menerapkan pengaturan blokir akses publik ke akun, pengaturan berlaku untuk semua Wilayah AWS secara global. Pengaturan tersebut mungkin tidak berlaku di semua Wilayah secara langsung atau bersamaan, tetapi pada akhirnya akan menyebar ke semua Wilayah.

Topik

- Pengaturan blokir akses publik
- Melakukan operasi akses publik blok pada titik akses
- Arti “publik”
- Menggunakan IAM Access Analyzer untuk S3 untuk meninjau bucket publik
- Izin
- Mengonfigurasi blokir akses publik
- Mengonfigurasi pengaturan blokir akses publik untuk akun Anda
- Mengonfigurasi pengaturan blokir akses publik untuk bucket S3 Anda

Pengaturan blokir akses publik

S3 Blokir Akses Publik menyediakan empat pengaturan. Anda dapat menerapkan pengaturan ini dalam kombinasi apa pun ke titik akses individu, bucket, atau seluruh Akun AWS. Jika Anda menerapkan pengaturan tersebut ke sebuah akun, pengaturan tersebut akan berlaku untuk semua bucket dan titik akses yang dimiliki oleh akun tersebut. Demikian pula, jika Anda menerapkan pengaturan tersebut pada bucket, maka pengaturan tersebut berlaku untuk semua titik akses yang terkait dengan bucket tersebut.

Tabel berikut berisi pengaturan-pengaturan yang tersedia.

Nama	Deskripsi
`BlockPublicAcls`	Mengatur opsi ini menjadi `TRUE` akan menyebabkan perilaku berikut ini: • Panggilan acl PUT bucket dan acl PUT Objek tidak berfungsi jika daftar kontrol akses (ACL) yang ditentukan bersifat publik. • Panggilan PUT Objek dianggap gagal jika permintaan mencakup ACL publik. • Jika pengaturan ini diterapkan ke akun, maka panggilan PUT bucket gagal jika permintaan menyertakan ACL publik. Saat pengaturan ini disetel ke`TRUE`, operasi yang ditentukan gagal (baik dilakukan melalui REST API AWS CLI, atau AWS SDK). Namun demikian, kebijakan dan ACL yang sudah ada untuk bucket dan objek tidak dimodifikasi. Pengaturan ini mengaktifkan Anda untuk melindungi dari akses publik sekaligus mengizinkan Anda untuk mengaudit, menyempurnakan, atau sebaliknya mengubah kebijakan dan ACL yang sudah ada untuk bucket dan objek Anda. **Note** Titik akses tidak memiliki ACL yang memiliki asosiasi dengannya . Jika Anda menerapkan pengaturan ini ke titik akses, maka pengaturan bertindak sebagai jalur ke bucket yang mendasarinya. Jika titik akses mengaktifkan pengaturan ini, maka permintaan yang dibuat melalui titik akses tersebut akan tetap berfungsi seolah-olah bucket yang mendasari akan mengaktifkan pengaturan ini, terlepas dari apakah bucket benar-benar mengaktifkan pengaturan tersebut atau tidak.
`IgnorePublicAcls`	Mengatur opsi ini menjadi `TRUE` akan menyebabkan Amazon S3 mengabaikan semua ACL publik di bucket dan setiap objek yang dimuatnya. Pengaturan ini mengaktifkan Anda untuk dengan aman melakukan blokir akses publik yang diberikan oleh ACL sekaligus masih mengizinkan

Nama	Deskripsi
	panggilan PUT Objek yang menyertakan ACL publik (berbeda dengan `BlockPublicAcls` , yang menolak panggilan PUT Objek yang menyertakan ACL publik). Pengaktifan pengaturan ini tidak memengaruhi ketahanan dari ACL yang ada dan tidak mencegah ACL publik baru diatur. **Note** Titik akses tidak memiliki ACL yang memiliki asosiasi dengannya . Jika Anda menerapkan pengaturan ini ke titik akses, maka pengaturan bertindak sebagai jalur ke bucket yang mendasarinya. Jika titik akses mengaktifkan pengaturan ini, maka permintaan yang dibuat melalui titik akses tersebut akan tetap berfungsi seolah-olah bucket yang mendasari akan mengaktifkan pengaturan ini, terlepas dari apakah bucket benar-benar mengaktifkan pengaturan tersebut atau tidak.

Nama	Deskripsi
`BlockPublicPolicy`	Pengaturan opsi ini ke bucket `TRUE` menyebabkan Amazon S3 menolak panggilan ke kebijakan PUT Bucket jika kebijakan bucket yang ditentukan mengizinkan akses publik. Pengaturan opsi `TRUE` ini menjadi bucket juga akan menyebabkan Amazon S3 menolak panggilan ke kebijakan PUT titik akses untuk semua titik akses akun bucket yang sama jika kebijakan yang ditentukan mengizinkan akses publik. Mengatur opsi ini untuk titik akses `TRUE` akan menyebabkan Amazon S3 menolak panggilan ke kebijakan PUT titik akses dan kebijakan PUT Bucket yang dibuat melalui titik akses tersebut jika kebijakan yang ditentukan (baik untuk titik akses atau bucket yang mendasari) memungkinkan akses publik. Anda dapat menggunakan pengaturan ini untuk memungkinkan pengguna mengelola kebijakan titik akses dan bucket tanpa mengizinkan mereka membagikan bucket atau objek yang dimuatnya secara publik. Mengaktif kan pengaturan ini tidak memengaruhi kebijakan titik akses atau bucket yang sudah ada. **Important** Untuk menggunakan pengaturan ini secara efektif, kami menyarank an Anda menerapkannya di level akun. Kebijakan bucket dapat memungkinkan pengguna untuk mengubah pengaturan blokir akses publik bucket. Oleh karena itu, pengguna yang memiliki izin untuk mengubah kebijakan bucket dapat memasukkan kebijakan yang memungkinkan mereka menonaktifkan pengaturan blokir akses publik untuk bucket tersebut. Jika pengaturan ini diaktifka n untuk seluruh akun, bukannya untuk bucket tertentu, Amazon S3 memblokir kebijakan publik bahkan jika pengguna mengubah kebijakan bucket untuk menonaktifkan pengaturan ini.

Nama	Deskripsi
`RestrictPublicBuckets`	Menyetel opsi ini untuk `TRUE` membatasi akses ke titik akses atau bucket dengan kebijakan publik hanya untuk prinsipal AWS layanan dan pengguna resmi dalam akun pemilik bucket dan akun pemilik jalur akses. Pengaturan ini memblokir semua akses lintas akun ke titik akses atau bucket (kecuali oleh prinsipal AWS layanan), sambil tetap memungkinkan pengguna di dalam akun untuk mengelola titik akses atau bucket. Mengaktifkan pengaturan ini tidak memengaruhi kebijakan titik akses atau bucket yang sudah ada, kecuali bahwa Amazon S3 memblokir akses publik dan akses lintas akun yang berasal dari setiap titik akses publik atau kebijakan bucket, termasuk delegasi non-publik ke akun tertentu.

Important

- Panggilan ke GET bucket dan GET Object selalu mengembalikan izin efektif yang ada untuk bucket atau objek yang ditentukan. Sebagai contoh, anggaplah sebuah bucket memiliki ACL yang memberikan akses publik, tetapi bucket tersebut juga mengaktifkan pengaturan `IgnorePublicAcls`. Dalam hal ini, acl GET bucket menghasilkan ACL yang mencerminkan izin akses yang diterapkan Amazon S3, bukannya ACL aktual yang terkait dengan bucket.
- Melakukan blok pengaturan akses publik tidak mengubah kebijakan atau ACL yang ada. Oleh karena itu, menghapus pengaturan blokir akses publik akan menyebabkan bucket atau objek yang memiliki kebijakan publik atau ACL dapat diakses oleh publik lagi.

Melakukan operasi akses publik blok pada titik akses

Untuk melakukan memblokir operasi akses publik pada titik akses, gunakan AWS CLI layanan`s3control`.

Important

Perhatikan bahwa saat ini tidak memungkinkan untuk mengubah pengaturan blokir akses publik milik titik akses setelah membuat titik akses. Dengan demikian, satu-satunya

cara untuk menentukan pengaturan blokir akses publik untuk titik akses adalah dengan menyertakannya saat membuat titik akses.

Arti “publik”

ACLs

Amazon S3 menganggap ACL bucket atau objek bersifat publik jika memberikan izin kepada anggota dari `AllUsers` atau grup `AuthenticatedUsers` yang telah ditentukan sebelumnya. Untuk informasi lebih lanjut tentang grup yang telah ditentukan sebelumnya, lihat Grup Amazon S3 yang sudah ditentukan sebelumnya.

Kebijakan bucket

Saat mengevaluasi kebijakan bucket, Amazon S3 memulai dengan mengasumsikan bahwa kebijakan tersebut bersifat publik. Itu kemudian mengevaluasi kebijakan tersebut untuk menentukan apakah kebijakan tersebut memenuhi syarat sebagai non-publik. Agar dianggap non-publik, suatu kebijakan bucket harus memberikan akses hanya ke nilai-nilai tetap (nilai yang tidak mengandung wildcard atau Variabel AWS Identity and Access Management Kebijakan) untuk satu atau lebih berikut ini:

- AWS Prinsipal, pengguna, peran, atau prinsipal layanan (mis.`aws:PrincipalOrgID`)
- Satu set Perutean Antar-Domain Tanpa Kelas (CIDRs), menggunakan `aws:SourceIp`. Untuk informasi lebih lanjut tentang CIDR, lihat RFC 4632 di situs web Editor RFC.

  > **Note**
  >
  > Kebijakan bucket yang memberikan akses yang dikondisikan pada kunci kondisi `aws:SourceIp` dengan rentang IP yang sangat luas (misalnya, 0.0.0.0/1) dievaluasi sebagai “publik.” Ini termasuk nilai yang lebih luas dari /8 untuk IPv4 dan /32 untuk IPv6 (tidak termasuk rentang pribadi RFC1918). Blokir akses publik akan menolak kebijakan “publik” ini dan mencegah akses lintas akun ke bucket yang sudah menggunakan kebijakan “publik: ini.

- `aws:SourceArn`
- `aws:SourceVpc`
- `aws:SourceVpce`
- `aws:SourceOwner`

- `aws:SourceAccount`
- `s3:x-amz-server-side-encryption-aws-kms-key-id`
- `aws:userid`, di luar dari pola "`AROLEID:*`"
- `s3:DataAccessPointArn`

> **Note**
>
> Ketika digunakan dalam kebijakan bucket, nilai ini dapat memuat wildcard untuk nama titik akses tanpa mengubah kebijakan publik, selama id akunnya tetap sama. Misalnya, memungkinkan akses ke `arn:aws:s3:us-west-2:123456789012:accesspoint/*` akan mengizinkan akses ke setiap titik akses yang terkait dengan akun `123456789012` di Wilayah `us-west-2`, tanpa membuat kebijakan bucket menjadi publik. Perhatikan bahwa perilaku ini berbeda untuk kebijakan titik akses. Untuk informasi selengkapnya, lihat Titik Akses.

- `s3:DataAccessPointAccount`

Untuk informasi lebih lanjut tentang kebijakan bucket, lihat Kebijakan bucket untuk Amazon S3.

Example : Kebijakan bucket publik

Menurut aturan ini, contoh kebijakan-kebijakan berikut ini dianggap publik.

```
{
  "Principal": "*",
  "Resource": "*",
  "Action": "s3:PutObject",
  "Effect": "Allow"
 }
```

```
{
  "Principal": "*",
  "Resource": "*",
  "Action": "s3:PutObject",
  "Effect": "Allow",
  "Condition": { "StringLike": {"aws:SourceVpc": "vpc-*"}}
 }
```

Anda dapat membuat kebijakan-kebijakan ini non-publik dengan memasukkan kunci kondisi yang tercantum sebelumnya, dengan menggunakan nilai tetap. Misalnya, Anda dapat membuat kebijakan terakhir sebelumnya menjadi non-publik dengan mengatur `aws:SourceVpc` dengan nilai tetap, seperti berikut ini.

```
{
  "Principal": "*",
  "Resource": "*",
  "Action": "s3:PutObject",
  "Effect": "Allow",
  "Condition": {"StringEquals": {"aws:SourceVpc": "vpc-91237329"}}
 }
```

Cara Amazon S3 mengevaluasi kebijakan bucket yang berisi pemberian akses publik dan non-publik

Contoh ini menunjukkan bagaimana Amazon S3 mengevaluasi kebijakan bucket yang berisi pemberian akses, baik publik maupun non-publik.

Anggaplah bucket memiliki kebijakan yang memberikan akses ke serangkaian pengguna utama tetap. Berdasarkan peraturan yang dijelaskan sebelumnya, kebijakan ini tidak boleh dibuat menjadi publik. Dengan demikian, jika Anda mengaktifkan pengaturan `RestrictPublicBuckets`, maka kebijakan tetap berlaku sebagaimana yang tertulis, karena `RestrictPublicBuckets` hanya berlaku untuk bucket yang memiliki kebijakan publik. Namun, jika Anda menambahkan pernyataan publik pada kebijakan tersebut, `RestrictPublicBuckets` berpengaruh pada bucket. Ini hanya memungkinkan kepala AWS layanan dan pengguna resmi akun pemilik bucket untuk mengakses bucket.

Sebagai contoh, anggaplah bahwa sebuah bucket yang dimiliki "Akun-1" memiliki kebijakan yang memuat hal-hal berikut:

1. Pernyataan yang memberikan akses ke AWS CloudTrail (yang merupakan prinsipal AWS layanan)
2. Pernyataan yang memberikan akses ke akun “Akun-2”
3. Pernyataan yang memberikan akses ke publik, misalnya dengan menentukan `"Principal": "*"` tanpa membatasi `Condition`

Kebijakan ini memenuhi kualifikasi sebagai kebijakan bersifat publik karena pernyataan ketiga. Dengan kebijakan ini diberlakukan dan `RestrictPublicBuckets` diaktifkan, Amazon S3 mengizinkan akses hanya dengan. CloudTrail Meskipun pernyataan 2 tidak bersifat publik, Amazon S3 nonaktifkan akses oleh "Akun-2." Hal ini karena pernyataan 3 menjadikan kebijakan

tersebut bersifat publik, sehingga `RestrictPublicBuckets` berlaku. Oleh karena itu, Amazon S3 menonaktifkan akses lintas akun, meskipun kebijakan tersebut mendelegasikan akses ke akun tertentu, yakni “Akun-2.” Tetapi jika Anda menghapus pernyataan 3 dari kebijakan tersebut, maka kebijakan tersebut tidak akan memenuhi syarat sebagai kebijakan bersifat publik, dan `RestrictPublicBuckets` tidak lagi berlaku. Dengan demikian, “Akun-2” mendapatkan kembali akses ke bucket, bahkan jika Anda membiarkan `RestrictPublicBuckets` tetap aktif.

Titik Akses

Amazon S3 mengevaluasi pengaturan blokir akses publik dengan sedikit berbeda untuk titik akses dibandingkan dengan bucket. Aturan yang diterapkan Amazon S3 untuk menentukan kapan kebijakan titik akses publik biasanya sama dengan titik akses untuk bucket, kecuali dalam situasi berikut:

- Titik akses yang memiliki asal jaringan VPC selalu dianggap sebagai non-publik, terlepas dari konten kebijakan titik aksesnya.
- Kebijakan titik akses yang memberikan akses ke serangkaian titik akses dengan menggunakan `s3:DataAccessPointArn` dianggap bersifat publik. Perhatikan bahwa perilaku ini berbeda dengan perilaku untuk kebijakan bucket. Misalnya, kebijakan bucket yang memberikan akses ke nilai `s3:DataAccessPointArn` yang sesuai dengan `arn:aws:s3:us-west-2:123456789012:accesspoint/*` tidak dianggap publik. Namun demikian, pernyataan yang sama dalam kebijakan titik akses akan menjadikan titik akses tersebut menjadi publik.

Menggunakan IAM Access Analyzer untuk S3 untuk meninjau bucket publik

Anda dapat menggunakan Penganalisis Akses untuk S3 untuk meninjau bucket dengan ACL bucket, kebijakan bucket, atau kebijakan titik akses yang memberikan akses publik. IAM Access Analyzer for S3 memberi tahu Anda tentang bucket yang dikonfigurasi untuk memungkinkan akses ke siapa pun di internet atau lainnya Akun AWS, termasuk Akun AWS di luar organisasi Anda. Untuk setiap bucket publik atau bucket bersama, Anda akan menerima temuan yang melaporkan sumber dan tingkat akses publik atau akses bersama.

Pada Penganalisis Akses untuk S3, Anda dapat memblokir semua akses publik ke bucket dengan sekali klik. Anda juga dapat menelusuri pengaturan izin tingkat bucket untuk mengonfigurasi tingkat akses granular. Untuk kasus penggunaan tertentu dan terverifikasi yang memerlukan akses publik atau akses bersama, Anda dapat menyatakan dan mencatat maksud Anda untuk bucket agar tetap sebagai bucket dengan akses publik atau akses bersama dengan mengarsipkan temuan untuk bucket tersebut.

Dalam peristiwa langka, Penganalisis Akses untuk S3 mungkin tidak melaporkan temuan untuk bucket, yang berdasarkan evaluasi blokir akses publik Amazon S3 dilaporkan sebagai publik. Hal ini terjadi karena blokir akses publik Amazon S3 meninjau kebijakan-kebijakan untuk tindakan saat ini dan setiap tindakan potensial yang mungkin ditambahkan di masa depan, yang menyebabkan bucket menjadi bersifat publik. Di sisi lain, IAM Access Analyzer untuk S3 hanya menganalisis tindakan saat ini yang ditentukan untuk layanan Amazon S3 dalam evaluasi status akses.

Untuk informasi selengkapnya tentang IAM Access Analyzer untuk Amazon S3, lihat Meninjau akses bucket menggunakan IAM Access Analyzer untuk S3.

Izin

Untuk menggunakan fitur-fitur yang dimiliki Blokir Akses Publik Amazon S3, Anda harus memiliki izin-izin berikut.

Operasi	Izin yang diperlukan
Status kebijakan bucket GET	`s3:GetBucketPolicyStatus`
Pengaturan Blokir Akses Publik bucket GET	`s3:GetBucketPublicAccessBlock`
Pengaturan Blokir Akses Publik bucket PUT	`s3:PutBucketPublicAccessBlock`
Pengaturan Blokir Akses Publik bucket DELETE	`s3:PutBucketPublicAccessBlock`
Pengaturan Blokir Akses Publik akun GET	`s3:GetAccountPublicAccessBlock`
Pengaturan Blokir Akses Publik akun PUT	`s3:PutAccountPublicAccessBlock`
Pengaturan Blokir Akses Publik akun DELETE	`s3:PutAccountPublicAccessBlock`
Pengaturan Blokir Akses Publik titik akses PUT	`s3:CreateAccessPoint`

> **Note**
>
> Operasi DELETE memerlukan izin yang sama dengan operasi PUT. Tidak ada izin terpisah untuk operasi DELETE.

Mengonfigurasi blokir akses publik

Untuk informasi selengkapnya tentang mengonfigurasi blokir akses publik untuk bucket Amazon S3 Anda Akun AWS dan Anda, lihat topik berikut.

- Mengonfigurasi pengaturan blokir akses publik untuk akun Anda
- Mengonfigurasi pengaturan blokir akses publik untuk bucket S3 Anda

Mengonfigurasi pengaturan blokir akses publik untuk akun Anda

Fitur Blokir Akses Publik Amazon S3 menyediakan pengaturan untuk titik akses, bucket, dan akun untuk membantu Anda mengelola akses publik ke sumber daya Amazon S3. Secara default, bucket, titik akses, dan objek yang baru tidak mengizinkan akses publik.

Untuk informasi selengkapnya, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

> **Note**
>
> Pengaturan tingkat akun mengesampingkan pengaturan pada objek individual. Mengkonfigurasi akun Anda untuk memblokir akses publik akan mengesampingkan pengaturan akses publik yang dibuat ke objek individual dalam akun Anda.

Anda dapat menggunakan konsol S3, AWS SDK AWS CLI, dan REST API untuk mengonfigurasi blokir setelan akses publik untuk semua bucket di akun Anda. Untuk informasi selengkapnya, lihat bagian di bawah ini.

Untuk mengonfigurasi pengaturan blokir akses publik untuk bucket Anda, lihat Mengonfigurasi pengaturan blokir akses publik untuk bucket S3 Anda. Untuk informasi lebih lanjut tentang titik akses, lihat Melakukan operasi akses publik blok pada titik akses.

Menggunakan konsol S3

Blokir akses publik Amazon S3 mencegah aplikasi dari pengaturan apa pun yang memungkinkan akses publik ke data dalam S3 bucket. Bagian ini menjelaskan cara mengedit pengaturan blokir akses publik untuk semua bucket S3 di Akun AWS Anda. Untuk informasi lebih lanjut tentang pemblokiran akses publik, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

Untuk mengedit blokir setelan akses publik untuk semua bucket S3 di Akun AWS

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pilih Blokir pengaturan Akses Publik untuk akun ini.
3. Memilih Edit untuk mengubah pengaturan blokir akses publik untuk semua bucket di Akun AWS Anda.
4. Pilih pengaturan yang ingin Anda ubah, lalu pilih Simpan perubahan.
5. Saat Anda diminta untuk mengonfirmasi, masukkan **`confirm`**. Kemudian, pilih Konfirmasi untuk menyimpan perubahan Anda.

Menggunakan AWS CLI

Anda dapat menggunakan Blokir Akses Publik Amazon S3 melalui AWS CLI. Untuk informasi selengkapnya tentang pengaturan dan penggunaan AWS CLI, lihat Apa itu AWS Command Line Interface?

Akun

Untuk melakukan operasi blokir akses publik pada akun, gunakan AWS CLI layanan `s3control`. Operasi tingkat akun yang menggunakan layanan ini adalah sebagai berikut:

- PUT PublicAccessBlock (untuk akun)
- GET PublicAccessBlock (untuk akun)
- HAPUS PublicAccessBlock (untuk akun)

Untuk informasi dan contoh tambahan, lihat put-public-access-blockdi AWS CLI Referensi.

Menggunakan AWS SDK

Java

Contoh berikut menunjukkan kepada Anda cara menggunakan Amazon S3 Block Public Access dengan AWS SDK for Java untuk menempatkan konfigurasi blok akses publik pada akun Amazon S3.

```
AWSS3ControlClientBuilder controlClientBuilder =
 AWSS3ControlClientBuilder.standard();
```

```
controlClientBuilder.setRegion(<region>);
controlClientBuilder.setCredentials(<credentials>);

AWSS3Control client = controlClientBuilder.build();
client.putPublicAccessBlock(new PutPublicAccessBlockRequest()
  .withAccountId(<account-id>)
  .withPublicAccessBlockConfiguration(new PublicAccessBlockConfiguration()
    .withIgnorePublicAcls(<value>)
    .withBlockPublicAcls(<value>)
    .withBlockPublicPolicy(<value>)
    .withRestrictPublicBuckets(<value>)));
```

> **Important**
>
> Contoh ini hanya berkaitan dengan operasi tingkat akun, yang menggunakan kelas pelanggan `AWSS3Control`. Untuk operasi tingkat bucket, lihat contoh sebelumnya.

Other SDKs

Untuk informasi tentang menggunakan AWS SDK lain, lihatMengembangkan dengan Amazon S3 menggunakan SDK AWS.

Penggunaan API REST

Untuk informasi tentang penggunaan Blokir Akses Publik Amazon S3 melalui API REST, lihat topik-topik berikut ini di Referensi API Amazon Simple Storage Service.

- Operasi tingkat akun
  - MENEMPATKAN PublicAccessBlock
  - DAPATKAN PublicAccessBlock
  - HAPUS PublicAccessBlock

Mengonfigurasi pengaturan blokir akses publik untuk bucket S3 Anda

Fitur Blokir Akses Publik Amazon S3 menyediakan pengaturan untuk titik akses, bucket, dan akun untuk membantu Anda mengelola akses publik ke sumber daya Amazon S3. Secara default, bucket, titik akses, dan objek yang baru tidak mengizinkan akses publik.

Untuk informasi selengkapnya, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

Anda dapat menggunakan konsol S3, AWS SDK AWS CLI, dan REST API untuk memberikan akses publik ke satu atau beberapa bucket. Anda juga dapat memblokir akses publik ke bucket yang sudah publik. Untuk informasi selengkapnya, lihat bagian di bawah ini.

Untuk mengonfigurasi pengaturan blokir akses publik untuk setiap bucket di akun Anda, lihat Mengonfigurasi pengaturan blokir akses publik untuk akun Anda. Untuk informasi tentang konfigurasi blokir akses publik untuk titik akses, lihat Melakukan operasi akses publik blok pada titik akses.

Menggunakan konsol S3

Blokir akses publik Amazon S3 mencegah aplikasi dari pengaturan apa pun yang memungkinkan akses publik ke data dalam S3 bucket. Bagian ini menjelaskan cara mengedit pengaturan Blokir Akses Publik untuk satu bucket S3 atau lebih. Untuk informasi tentang memblokir akses publik menggunakan AWS CLI, AWS SDK, dan Amazon S3 REST API, lihat. Melakukan blok akses publik ke penyimpanan Amazon S3 Anda

Anda dapat melihat apakah bucket Anda dapat diakses oleh publik di daftar Bucket. Di kolom Akses, Amazon S3 melabeli izin untuk bucket sebagai berikut:

- Public–Setiap orang memiliki akses ke satu atau lebih dari yang berikut ini: Buat daftar objek, Tulis objek, Baca dan izin tulis.
- Objek dapat bersifat publik–Bucket bukanlah informasi publik, tetapi siapa pun dengan izin yang sesuai dapat memberikan akses publik ke objek.
- Bucket dan objek tidak untuk umum–Bucket dan objek tidak memiliki akses publik.
- Hanya pengguna yang berwenang dari akun ini — Akses diisolasi untuk pengguna IAM dan peran dalam akun dan prinsip AWS layanan ini karena ada kebijakan yang memberikan akses publik.

Anda juga dapat memfilter pencarian bucket menurut jenis akses. Pilih jenis akses dari daftar menurun di samping bilah Cari bucket.

Jika Anda melihat `Error` saat Anda mencantumkan bucket dan pengaturan akses publiknya, Anda mungkin tidak memiliki izin yang diperlukan. Periksa untuk memastikan Anda memiliki izin berikut yang ditambahkan ke pengguna atau kebijakan peran Anda:

```
s3:GetAccountPublicAccessBlock
s3:GetBucketPublicAccessBlock
s3:GetBucketPolicyStatus
```

```
s3:GetBucketLocation
s3:GetBucketAcl
s3:ListAccessPoints
s3:ListAllMyBuckets
```

Dalam beberapa kasus yang jarang terjadi, permintaan juga dapat gagal karena gangguan Wilayah AWS .

Untuk mengedit pengaturan blokir akses publik Amazon S3 untuk satu bucket S3

Ikuti langkah-langkah ini jika Anda perlu mengubah pengaturan akses publik untuk bucket S3 tunggal.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, memilih nama bucket yang Anda inginkan.
3. Pilih Izin.
4. Pilih Edit untuk mengubah pengaturan akses publik untuk bucket. Untuk informasi selengkapnya tentang empat Peraturan Blokir Akses Publik Amazon S3, lihat Pengaturan blokir akses publik.
5. Memilih pengaturan yang ingin Anda ubah, lalu memilih Simpan.
6. Saat Anda diminta untuk mengonfirmasi, masukkan **`confirm`**. Kemudian, pilih Konfirmasi untuk menyimpan perubahan Anda.

Anda dapat mengubah pengaturan Blokir Akses Publik Amazon S3 saat Anda membuat bucket. Untuk informasi selengkapnya, lihat Membuat bucket.

Menggunakan AWS CLI

Untuk memblokir akses publik pada bucket atau untuk menghapus blok akses publik, gunakan AWS CLI layanan ini`s3api`. Operasi tingkat bucket yang menggunakan layanan ini adalah sebagai berikut:

- PUT PublicAccessBlock (untuk ember)
- GET PublicAccessBlock (untuk ember)
- DELETE PublicAccessBlock (untuk ember)
- DAPATKAN BucketPolicyStatus

Untuk informasi dan contoh selengkapnya, lihat put-public-access-blockdi AWS CLI Referensi.

Menggunakan AWS SDK

Java

```
AmazonS3 client = AmazonS3ClientBuilder.standard()
   .withCredentials(<credentials>)
   .build();

client.setPublicAccessBlock(new SetPublicAccessBlockRequest()
  .withBucketName(<bucket-name>)
  .withPublicAccessBlockConfiguration(new PublicAccessBlockConfiguration()
    .withBlockPublicAcls(<value>)
    .withIgnorePublicAcls(<value>)
    .withBlockPublicPolicy(<value>)
    .withRestrictPublicBuckets(<value>)));
```

> **Important**
>
> Contoh ini hanya berkaitan dengan operasi tingkat bucket, yang menggunakan kelas pelanggan `AmazonS3`. Untuk operasi tingkat akun, lihat contoh berikut.

Other SDKs

Untuk informasi tentang menggunakan AWS SDK lain, lihat Mengembangkan dengan Amazon S3 menggunakan SDK AWS.

Penggunaan API REST

Untuk informasi tentang penggunaan Blokir Akses Publik Amazon S3 melalui API REST, lihat topik-topik berikut ini di Referensi API Amazon Simple Storage Service.

- Operasi tingkat bucket
  - MENEMPATKAN PublicAccessBlock
  - DAPATKAN PublicAccessBlock
  - HAPUS PublicAccessBlock
  - DAPATKAN BucketPolicyStatus

Meninjau akses bucket menggunakan IAM Access Analyzer untuk S3

IAM Access Analyzer for S3 memberi tahu Anda tentang bucket S3 yang dikonfigurasi untuk memungkinkan akses ke siapa pun di internet atau lainnya Akun AWS, termasuk di luar organisasi Anda. Akun AWS Untuk setiap bucket publik atau kelompok bersama, Anda menerima temuan ke dalam sumber dan tingkat akses publik atau bersama. Misalnya, IAM Access Analyzer untuk S3 mungkin menunjukkan bahwa suatu bucket telah membaca atau menulis akses yang disediakan melalui daftar kontrol akses (ACL) bucket, kebijakan titik akses Multi-Wilayah, atau kebijakan titik akses. Dengan temuan ini, Anda dapat segera mengambil tindakan korektif yang tepat untuk memulihkan akses bucket ke apa yang Anda inginkan.

Saat meninjau bucket berisiko dalam Penganalisis Akses IAM untuk S3, Anda dapat memblokir semua akses publik ke bucket dengan sekali klik. Kami menyarankan agar Anda memblokir semua akses ke bucket Anda, kecuali jika Anda memerlukan akses publik untuk mendukung kasus penggunaan tertentu. Sebelum memblokir semua akses publik, pastikan bahwa aplikasi Anda akan terus berfungsi dengan benar tanpa akses publik. Untuk informasi selengkapnya, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

Anda juga dapat menelusuri pengaturan izin tingkat bucket untuk mengonfigurasi tingkat akses granular. Untuk kasus penggunaan kustom dan terverifikasi yang memerlukan akses publik, seperti hosting situs web statis, unduhan publik, atau berbagi akun silang, Anda dapat mengenali dan mencatat maksud Anda agar bucket tetap publik atau berbagi dengan mengarsipkan temuan untuk bucket. Anda dapat mempertahankan dan memodifikasi konfigurasi bucket ini kapan saja. Anda juga dapat mengunduh temuan Anda dalam bentuk laporan CSV untuk keperluan audit.

IAM Access Analyzer untuk S3 tersedia tanpa biaya tambahan, yang berada di konsol Amazon S3. Penganalisis Akses IAM untuk S3 didukung oleh Penganalisis Akses IAM (IAM) AWS Identity and Access Management . Untuk menggunakan IAM Access Analyzer untuk S3 di konsol Amazon S3, Anda harus mengunjungi konsol IAM dan mengaktifkan IAM Access Analyzer pada basis Per-wilayah.

Untuk informasi selengkapnya tentang IAM Access Analyzer, lihat Apa itu IAM Access Analyzer? di Panduan Pengguna IAM. Untuk informasi lebih lanjut tentang IAM Access Analyzer untuk S3, tinjau bagian berikut.

> **Important**
>
> - IAM Access Analyzer untuk S3 membutuhkan penganalisis tingkat akun. Untuk menggunakan IAM Access Analyzer untuk S3, Anda harus mengunjungi IAM Access

Analyzer dan membuat analyzer yang memiliki akun sebagai zona kepercayaan. Untuk informasi selengkapnya, lihat Mengaktifkan IAM Access Analyzer di Panduan Pengguna IAM.

- IAM Access Analyzer untuk S3 tidak menganalisis kebijakan titik akses yang dilampirkan ke titik akses lintas akun. Perilaku ini terjadi karena titik akses dan kebijakannya berada di luar zona kepercayaan, yaitu akun. Bucket yang mendelegasikan akses ke titik akses lintas akun tercantum di bawah Bucket dengan akses publik jika Anda belum menerapkan pengaturan blokir akses publik `RestrictPublicBuckets` ke bucket atau akun. Saat Anda menerapkan setelan `RestrictPublicBuckets` blokir akses publik, bucket dilaporkan di bawah Bucket dengan akses dari orang lain Akun AWS — termasuk pihak ketiga Akun AWS.
- Ketika kebijakan bucket atau ACL bucket ditambahkan atau diubah, IAM Access Analyzer menghasilkan dan memperbarui temuan berdasarkan perubahan dalam waktu 30 menit. Temuan terkait dengan pengaturan akses publik blok tingkat akun tidak dapat dibuat atau diperbarui hingga 6 jam setelah Anda mengubah pengaturan. Temuan yang terkait dengan Titik Akses Multi-Wilayah mungkin tidak dibuat atau diperbarui hingga enam jam setelah Titik Akses Multi-Wilayah dibuat, dihapus, atau Anda mengubah kebijakannya.

Topik

- Informasi apa yang disediakan IAM Access Analyzer untuk S3?
- Mengaktifkan IAM Access Analyzer untuk S3
- Memblokir semua akses publik
- Meninjau dan mengubah akses bucket
- Mengarsipkan temuan bucket
- Mengaktifkan temuan bucket yang diarsipkan
- Melihat detail temuan
- Mengunduh IAM Access Analyzer untuk laporan S3

Informasi apa yang disediakan IAM Access Analyzer untuk S3?

Penganalisis Akses untuk S3 menyediakan temuan untuk bucket yang dapat diakses di luar Akun AWS Anda. Bucket yang terdaftar Bucket dengan akses publik dapat diakses oleh siapa pun di internet. Jika Access Analyzer untuk S3 mengidentifikasi bucket publik, Anda juga melihat peringatan

di bagian atas halaman yang menunjukkan jumlah bucket publik di Wilayah Anda. Bucket yang terdaftar di bawah Bucket dengan akses dari orang lain Akun AWS — termasuk pihak ketiga Akun AWS dibagikan secara kondisional dengan orang lain Akun AWS, termasuk akun di luar organisasi Anda.

Untuk setiap bucket, IAM Access Analyzer untuk S3 memberikan informasi berikut:

- Nama Bucket
- Ditemukan oleh Access Analyzer - Saat IAM Access Analyzer untuk S3 menemukan akses bucket publik atau bersama.
- Dibagikan melalui - Cara bucket dibagikan—melalui kebijakan bucket, ACL bucket, kebijakan Titik Akses Multi-Wilayah, atau kebijakan titik akses. Titik Akses Multi-Wilayah dan titik akses lintas akun ditampilkan di bawah titik akses. Bucket dapat dibagi melalui kebijakan dan ACL. Jika ingin menemukan dan meninjau sumber untuk akses bucket Anda, Anda dapat menggunakan informasi dalam kolom ini sebagai titik awal untuk mengambil tindakan korektif yang cepat dan tepat.
- Status - Status temuan bucket. IAM Access Analyzer untuk S3 menampilkan temuan untuk semua bucket publik dan bersama.
  - Aktif - Temuan belum ditinjau.
  - Diarsipkan - Temuan telah ditinjau dan dikonfirmasi sesuai tujuan.
  - Semua - Semua temuan untuk ember yang bersifat publik atau dibagikan dengan orang lain Akun AWS, termasuk Akun AWS di luar organisasi Anda.
- Tingkat akses - Izin akses diberikan untuk bucket:
  - Daftar - Cantumkan sumber daya.
  - Baca - Membaca, tetapi tidak mengedit konten dan atribut sumber daya.
  - Tulis - Membuat, menghapus, atau memodifikasi sumber daya.
  - Izin - Berikan atau ubah izin sumber daya.
  - Penandaan - Perbarui tag yang terkait dengan sumber daya.

Mengaktifkan IAM Access Analyzer untuk S3

Untuk menggunakan IAM Access Analyzer untuk S3, Anda harus menyelesaikan langkah prasyarat berikut.

1. Berikan izin yang diperlukan.

Untuk informasi lebih lanjut, lihat Izin yang Diperlukan untuk menggunakan Access Analyzer dalam Panduan Pengguna IAM.

2. Kunjungi IAM untuk membuat penganalisis level akun untuk setiap Wilayah tempat Anda ingin menggunakan Access Analyzer.

   IAM Access Analyzer untuk S3 membutuhkan penganalisis tingkat akun. Untuk menggunakan Access Analyzer untuk S3, Anda harus membuat penganalisis yang memiliki akun sebagai zona kepercayaan. Untuk informasi selengkapnya, lihat Mengaktifkan IAM Access Analyzer di Panduan Pengguna IAM.

Memblokir semua akses publik

Jika Anda ingin memblokir semua akses ke bucket dalam satu klik, Anda dapat menggunakan Blokir semua akses publik dalam Access Analyzer untuk S3. Ketika Anda memblokir semua akses publik ke bucket, akses publik tidak diberikan. Kami menyarankan agar Anda memblokir semua akses publik ke bucket Anda kecuali jika Anda memerlukan akses publik untuk mendukung kasus penggunaan tertentu yang terverifikasi. Sebelum memblokir semua akses publik, pastikan bahwa aplikasi Anda akan terus berfungsi dengan benar tanpa akses publik.

Jika tidak ingin memblokir semua akses publik ke bucket Anda, Anda dapat mengedit pengaturan akses publik blok di konsol Amazon S3 untuk mengonfigurasi tingkat akses granular ke bucket Anda. Untuk informasi selengkapnya, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

Dalam peristiwa langka, Penganalisis Akses untuk S3 mungkin tidak melaporkan temuan untuk bucket, yang berdasarkan evaluasi blokir akses publik Amazon S3 dilaporkan sebagai publik. Hal ini terjadi karena blokir akses publik Amazon S3 meninjau kebijakan-kebijakan untuk tindakan saat ini dan setiap tindakan potensial yang mungkin ditambahkan di masa depan, yang menyebabkan bucket menjadi bersifat publik. Di sisi lain, IAM Access Analyzer untuk S3 hanya menganalisis tindakan saat ini yang ditentukan untuk layanan Amazon S3 dalam evaluasi status akses.

Untuk memblokir semua akses publik ke bucket menggunakan Access Analyzer IAM untuk S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi di sebelah kiri, di bawah Dasbor, pilih Access analyzer untuk S3.
3. Di IAM Access Analyzer untuk S3, pilih bucket.
4. Pilih Blokir semua akses publik.

5. Untuk mengonfirmasi maksud Anda untuk memblokir semua akses publik ke bucket, di Blokir semua akses publik (pengaturan bucket), masukkan **`confirm`**.

   Amazon S3 memblokir semua akses publik ke bucket Anda. Status pembaruan temuan bucket untuk diselesaikan, dan bucket menghilang dari daftar Access Analyzer untuk S3. Jika Anda ingin meninjau bucket yang diselesaikan, buka IAM Access Analyzer di Konsol IAM.

Meninjau dan mengubah akses bucket

Jika Anda tidak bermaksud memberikan akses ke publik atau lainnya Akun AWS, termasuk akun di luar organisasi, Anda dapat mengubah ACL bucket, kebijakan bucket, kebijakan Titik Akses Multi-Wilayah, atau kebijakan jalur akses untuk menghapus akses ke bucket. Dibagikan melalui kolom menunjukkan semua sumber akses bucket: kebijakan ACL bucket, tujuan persetujuan, dan/atau kebijakan titik akses. Titik Akses Multi-Wilayah dan titik akses lintas akun ditampilkan di bawah titik akses.

Untuk meninjau dan mengubah kebijakan bucket, bucket ACL, Titik Akses Multi-Wilayah, atau kebijakan titik akses

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi, pilih Access analyzer untuk S3.
3. Untuk melihat apakah akses publik atau akses bersama diberikan melalui kebijakan bucket, kebijakan Titik Akses Multi-Wilayah, atau kebijakan titik akses, lihat di kolom Dibagikan melalui.
4. Di bagian bawah Bucket, pilih nama bucket dengan kebijakan bucket, kebijakan bucket ACL, atau kebijakan titik akses yang ingin Anda ubah atau tinjau.
5. Jika Anda ingin mengubah atau melihat bucket ACL:

   a. Pilih Izin.
   b. Pilih Daftar Kontrol Akses.
   c. Tinjau bucket ACL Anda, dan buat perubahan sesuai kebutuhan.

      Untuk informasi selengkapnya, lihat Mengonfigurasi ACL.
6. Jika Anda ingin mengubah atau meninjau kebijakan bucket:

   a. Pilih Izin.
   b. Pilih Kebijakan Bucket.
   c. Tinjau atau ubah kebijakan bucket sesuai kebutuhan.

Untuk informasi selengkapnya, lihat Menambahkan kebijakan bucket dengan menggunakan konsol Amazon S3.

7. Jika Anda ingin mengubah atau melihat kebijakan Titik Akses Multi-Wilayah:

   a. Pilih Titik Akses Multi-Wilayah.

   b. Pilih nama Titik Akses Multi-Wilayah.

   c. Tinjau atau ubah kebijakan Titik Akses Multi-Wilayah sesuai kebutuhan.

   Untuk informasi selengkapnya, lihat Izin.

8. Jika Anda ingin meninjau atau mengubah kebijakan titik akses:

   a. Memilih titik akses.

   b. Pilih nama titik akses.

   c. Tinjau atau ubah akses sesuai kebutuhan.

   Untuk informasi selengkapnya, lihat Menggunakan titik akses Amazon S3 dengan konsol Amazon S3.

Jika Anda mengedit atau menghapus Bacbu ACL, kebijakan bucket, atau kebijakan titik akses untuk menghapus akses publik atau akses bersama, status pembaruan temuan bucket untuk diselesaikan. Temuan bucket yang diselesaikan menghilang dari IAM Access Analyzer untuk daftar S3, tetapi Anda dapat melihatnya di IAM Access Analyzer.

Mengarsipkan temuan bucket

Jika bucket memberikan akses ke publik atau lainnya Akun AWS, termasuk akun di luar organisasi Anda, untuk mendukung kasus penggunaan tertentu (misalnya, situs web statis, unduhan publik, atau berbagi lintas akun), Anda dapat mengarsipkan temuan untuk bucket. Saat Anda mengarsipkan temuan bucket, Anda menyatakan dan mencatat maksud Anda agar bucket tetap bersifat publik, atau berbagi. Temuan bucket yang diarsipkan tetap ada dalam daftar Access Analyzer untuk S3 sehingga Anda selalu mengetahui bucket mana yang publik atau dibagikan.

Untuk mengarsipkan temuan bucket dalam IAM Access Analyzer untuk S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi, pilih Access analyzer untuk S3.

3. Access Analyzer IAM untuk S3, pilih bucket aktif.
4. Untuk mengetahui maksud Anda agar bucket ini dapat diakses oleh publik atau lainnya Akun AWS, termasuk akun di luar organisasi Anda, pilih Arsip.
5. Masukkan **`confirm`**, dan pilih Arsipkan.

Mengaktifkan temuan bucket yang diarsipkan

Setelah mengarsipkan temuan, Anda dapat selalu mempertahankan temuan tersebut dan mengubah status kembali menjadi aktif, yang menunjukkan bahwa bucket memerlukan peninjauan lain.

Untuk mengaktifkan temuan bucket yang diarsipkan di IAM Access Analyzer untuk S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi, pilih Access analyzer untuk S3.
3. Pilih bucket findings yang diarsipkan.
4. Pilih Tandai sebagai aktif.

Melihat detail temuan

Jika Anda perlu melihat informasi selengkapnya tentang bucket, Anda dapat membuka bucket untuk menemukan detail di IAM Access Analyzer di Konsol IAM.

Untuk melihat detail temuan dalam IAM Access Analyzer untuk S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi, pilih Access analyzer untuk S3.
3. Di IAM Access Analyzer untuk S3, pilih bucket.
4. Pilih Lihat detail.

   Detail temuan terbuka di IAM Access Analyzer di Konsol IAM.

Mengunduh IAM Access Analyzer untuk laporan S3

Anda dapat mengunduh temuan bucket Anda sebagai laporan CSV yang dapat Anda gunakan untuk tujuan audit. Laporan mencakup informasi yang sama yang Anda lihat dalam Access Analyzer untuk S3 pada konsol Amazon S3.

Untuk mengunduh laporan

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi di sebelah kiri, pilih Access analyzer untuk S3.
3. Pada filter Area, pilih Area.

   Access Analyzer IAM untuk pembaruan S3 untuk menunjukkan bucket untuk Area yang dipilih.
4. Pilih Unduh laporan.

   Laporan CSV dibuat dan disimpan ke komputer Anda.

Memverifikasi kepemilikan bucket dengan syarat pemilik bucket

Kondisi pemilik bucket Amazon S3 memastikan bahwa bucket yang Anda gunakan dalam operasi S3 adalah milik yang Anda harapkan. Akun AWS

Sebagian besar operasi S3 membaca atau menulis ke bucket S3 tertentu. Operasi ini meliputi pengunggahan, penyalinan, dan pengunduhan objek, pengambilan atau modifikasi konfigurasi bucket, dan pengambilan atau modifikasi konfigurasi objek. Saat Anda melakukan operasi ini, Anda menentukan bucket yang ingin Anda gunakan dengan menyertakan nama dengan permintaan. Misalnya, untuk mengambil objek dari S3, Anda membuat permintaan yang menentukan nama bucket dan kunci objek untuk diambil dari bucket tersebut.

Karena Amazon S3 mengidentifikasi bucket berdasarkan namanya, aplikasi yang menggunakan nama bucket yang salah dalam permintaan dapat secara tidak sengaja melakukan operasi terhadap bucket yang berbeda dari yang diharapkan. Untuk membantu menghindari interaksi bucket yang tidak disengaja dalam situasi seperti ini, Anda dapat menggunakan ketentuan pemilik bucket. Syarat pemilik bucket mengaktifkan Anda memverifikasi bahwa bucket target dimiliki oleh Akun AWS yang diharapkan, sehingga memberikan lapisan jaminan tambahan bahwa operasi S3 Anda memiliki efek yang Anda inginkan.

Topik

- Kapan harus menggunakan ketentuan pemilik bucket
- Memverifikasi pemilik bucket
- Contoh
- Pembatasan dan batasan

Kapan harus menggunakan ketentuan pemilik bucket

Kami merekomendasikan penggunaan ketentuan pemilik bucket setiap kali Anda melakukan operasi S3 yang didukung dan mengetahui ID akun pemilik bucket yang diharapkan. Ketentuan pemilik bucket tersedia untuk semua operasi S3 Object dan sebagian besar operasi bucket S3. Untuk daftar operasi S3 yang tidak mendukung ketentuan pemilik bucket, lihat Pembatasan dan batasan.

Untuk melihat manfaat menggunakan kondisi pemilik bucket, pertimbangkan skenario berikut yang melibatkan AWS pelanggan Bea:

1. Bea mengembangkan aplikasi yang menggunakan Amazon S3. Selama pengembangan, Bea menggunakan pengujiannya hanya Akun AWS untuk membuat bucket bernama`bea-data-test`, dan mengonfigurasi aplikasinya untuk membuat permintaan. `bea-data-test`
2. Bea men-deploy aplikasinya, tetapi lupa mengonfigurasi ulang aplikasi untuk menggunakan bucket dalam Akun AWS produksinya.
3. Dalam produksi, aplikasi Bea membuat permintaan ke `bea-data-test`, yang berhasil. Hal ini menyebabkan data produksi ditulis ke bucket dalam akun pengujian Bea.

Bea dapat membantu melindungi dari situasi seperti ini dengan menggunakan ketentuan pemilik bucket. Dengan kondisi pemilik bucket, Bea dapat menyertakan Akun AWS ID pemilik bucket yang diharapkan dalam permintaannya. Amazon S3 kemudian memeriksa ID akun pemilik bucket sebelum memproses setiap permintaan. Apabila pemilik bucket yang sesungguhnya tidak sesuai dengan pemilik bucket yang diharapkan, permintaan gagal.

Jika Bea menggunakan ketentuan pemilik bucket, skenario yang dijelaskan sebelumnya tidak akan mengakibatkan aplikasi Bea secara tidak sengaja menulis ke bucket pengujian. Sebagai gantinya, permintaan yang dibuat aplikasinya pada langkah 3 akan gagal dengan pesan kesalahan `Access Denied`. Dengan menggunakan syarat pemilik bucket, Bea membantu menghilangkan risiko interaksi yang tidak disengaja dengan bucket dalam Akun AWS yang salah.

Memverifikasi pemilik bucket

Untuk menggunakan ketentuan pemilik bucket, Anda menyertakan parameter dengan permintaan Anda yang menentukan pemilik bucket yang diharapkan. Sebagian besar operasi S3 hanya melibatkan satu bucket, dan hanya memerlukan satu parameter ini untuk menggunakan ketentuan pemilik bucket. Untuk operasi `CopyObject`, parameter pertama ini menentukan pemilik bucket tujuan yang diharapkan, dan Anda menyertakan parameter kedua untuk menentukan pemilik bucket sumber yang diharapkan.

Saat Anda membuat permintaan yang menyertakan parameter ketentuan pemilik bucket, S3 memeriksa ID akun pemilik bucket terhadap parameter yang ditentukan sebelum memproses permintaan. Jika parameter cocok dengan ID akun pemilik bucket, S3 akan memproses permintaan. Jika parameter tidak cocok dengan ID akun pemilik bucket, permintaan akan gagal dengan pesan kesalahan `Access Denied`.

Anda dapat menggunakan kondisi pemilik bucket dengan AWS Command Line Interface (AWS CLI), AWS SDK, dan Amazon S3 REST API. Saat menggunakan kondisi pemilik bucket dengan API REST Amazon S3 AWS CLI dan Amazon S3, gunakan nama parameter berikut.

Metode akses	Parameter untuk operasi non penyalinan	Parameter sumber operasi penyalinan	Parameter tujuan operasi penyalinan
AWS CLI	`--expected-bucket-owner`	`--expected-source-bucket-owner`	`--expected-bucket-owner`
API REST Amazon S3	Header `x-amz-expected-bucket-owner`	Header `x-amz-source-expected-bucket-owner`	Header `x-amz-expected-bucket-owner`

Nama parameter yang diperlukan untuk menggunakan syarat pemilik bucket dengan SDK AWS akan berbeda-beda tergantung bahasanya. Untuk menentukan parameter yang diperlukan, lihat dokumentasi SDK untuk bahasa yang Anda inginkan. Anda dapat menemukan dokumentasi SDK di Alat untuk Membangun di AWS.

Contoh

Contoh berikut menunjukkan bagaimana Anda dapat menerapkan kondisi pemilik bucket di Amazon S3 menggunakan AWS CLI atau. AWS SDK for Java 2.x

Example

Contoh: Mengunggah sebuah objek

Contoh berikut menunjukkan cara unggah sebuah objek ke bucket S3 *`example-s3-bucket1`*, menggunakan syarat pemilik bucket untuk memastikan bahwa *`example-s3-bucket1`* dimiliki oleh Akun AWS `111122223333`.

AWS CLI

```
aws s3api put-object \
                 --bucket example-s3-bucket1 --key exampleobject --
body example_file.txt \
                 --expected-bucket-owner 111122223333
```

AWS SDK for Java 2.x

```
public void putObjectExample() {
    S3Client s3Client = S3Client.create();;
    PutObjectRequest request = PutObjectRequest.builder()
            .bucket("example-s3-bucket1")
            .key("exampleobject")
            .expectedBucketOwner("111122223333")
            .build();
    Path path = Paths.get("example_file.txt");
    s3Client.putObject(request, path);
}
```

Example

Contoh: Menyalin sebuah objek

Contoh berikut menyalin objek `object1` dari bucket S3 *example-s3-bucket1* ke bucket S3 *example-s3-bucket2*. Itu menggunakan syarat pemilik bucket untuk memastikan bahwa bucket dimiliki oleh akun yang diharapkan sesuai dengan tabel berikut.

Bucket	Pemilik yang diharapkan
example-s3-bucket1	111122223333
example-s3-bucket2	444455556666

AWS CLI

```
aws s3api copy-object --copy-source example-s3-bucket1/object1 \
                            --bucket example-s3-bucket2 --key object1copy \
```

```
                                --expected-source-bucket-owner 111122223333 --expected-
bucket-owner 444455556666
```

AWS SDK for Java 2.x

```
public void copyObjectExample() {
        S3Client s3Client = S3Client.create();
        CopyObjectRequest request = CopyObjectRequest.builder()
                .copySource("example-s3-bucket1/object1")
                .destinationBucket("example-s3-bucket2")
                .destinationKey("object1copy")
                .expectedSourceBucketOwner("111122223333")
                .expectedBucketOwner("444455556666")
                .build();
        s3Client.copyObject(request);
    }
```

Example

Contoh: Mengambil kebijakan bucket

Contoh berikut ini mengambil kebijakan akses untuk bucket S3 *example-s3-bucket1*, menggunakan syarat pemilik bucket untuk memastikan bahwa *example-s3-bucket1* dimiliki oleh Akun AWS 111122223333.

AWS CLI

```
aws s3api get-bucket-policy --bucket example-s3-bucket1 --expected-bucket-
owner 111122223333
```

AWS SDK for Java 2.x

```
public void getBucketPolicyExample() {
    S3Client s3Client = S3Client.create();
    GetBucketPolicyRequest request = GetBucketPolicyRequest.builder()
            .bucket("example-s3-bucket1")
            .expectedBucketOwner("111122223333")
            .build();
    try {
        GetBucketPolicyResponse response = s3Client.getBucketPolicy(request);
    }
```

```
    catch (S3Exception e) {
        // The call was transmitted successfully, but Amazon S3 couldn't process
        // it, so it returned an error response.
        e.printStackTrace();
    }
}
```

Pembatasan dan batasan

Ketentuan pemilik bucket Amazon S3 memiliki pembatasan dan batasan berikut:

- Nilai parameter kondisi pemilik bucket harus berupa Akun AWS ID (nilai numerik 12 digit). Pengguna utama layanan tidak didukung.
- Kondisi pemilik bucket tidak tersedia untuk CreateBucket, ListBuckets, atau operasi apa pun yang termasuk dalam Kontrol AWS S3. Amazon S3 mengabaikan parameter ketentuan pemilik bucket yang disertakan dengan permintaan ke operasi ini.
- Ketentuan pemilik bucket hanya memverifikasi bahwa akun yang ditentukan dalam parameter verifikasi memiliki bucket. Ketentuan pemilik bucket tidak memeriksa konfigurasi bucket. Ini juga tidak menjamin bahwa konfigurasi bucket memenuhi ketentuan tertentu atau sesuai untuk status sebelumnya.

Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda

Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda dan untuk menonaktifkan atau mengaktifkan daftar kontrol akses (ACL). Secara default, Kepemilikan Objek disetel ke pengaturan diberlakukan pemilik Bucket dan semua ACL dinonaktifkan. Ketika ACL dinonaktifkan, pemilik bucket memiliki semua objek di bucket dan mengelola akses ke data secara eksklusif menggunakan kebijakan manajemen akses.

Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL, dan kami menyarankan agar Anda tetap menonaktifkan ACL kecuali dalam keadaan yang tidak biasa di mana Anda harus mengontrol akses untuk setiap objek satu per satu. Dengan ACL dinonaktifkan, Anda dapat menggunakan kebijakan untuk mengontrol akses ke setiap objek di bucket dengan lebih mudah, terlepas dari siapa yang mengunggah objek di bucket Anda.

Kepemilikan Objek memiliki tiga pengaturan yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda, dan untuk menonaktifkan atau mengaktifkan ACL:

ACL dinonaktifkan

- Pemilik bucket yang diberlakukan (default)–ACL dinonaktifkan, dan pemilik bucket secara otomatis memilikinya, serta mendapatkan kontrol penuh atas setiap objek di dalam bucket. ACL tidak lagi memengaruhi izin terhadap data di bucket S3. Bucket tersebut menggunakan kebijakan untuk menentukan kontrol akses.

ACL diaktifkan

- Pemilik bucket yang dipilih—Pemilik bucket memiliki dan diberikan kendali penuh atas objek baru yang ditulis akun lain ke bucket dengan ACL `bucket-owner-full-control` yang dibatasi.
- Penulis objek– Akun AWS yang mengunggah objek memiliki objek tersebut, mempunyai kontrol penuh atas objek tersebut, dan dapat memberikan akses kepada pengguna lain melalui ACL.

Untuk sebagian besar kasus penggunaan modern di S3, sebaiknya Anda menonaktifkan ACL dengan menerapkan pengaturan yang diberlakukan pemilik Bucket dan menggunakan kebijakan bucket Anda untuk berbagi data dengan pengguna di luar akun Anda sesuai kebutuhan. Pendekatan ini menyederhanakan manajemen izin. Anda dapat menonaktifkan ACL pada bucket yang baru dibuat dan yang sudah ada. Untuk bucket yang baru dibuat, ACL dinonaktifkan secara default. Dalam kasus bucket yang sudah ada yang sudah memiliki objek di dalamnya, setelah Anda menonaktifkan ACL, ACL objek dan bucket tidak lagi menjadi bagian dari evaluasi akses, dan akses diberikan atau ditolak berdasarkan kebijakan. Untuk bucket yang ada, Anda dapat mengaktifkan kembali ACL kapan saja setelah menonaktifkannya, dan bucket dan ACL objek yang sudah ada sebelumnya dipulihkan.

Sebelum menonaktifkan ACL, sebaiknya tinjau kebijakan bucket untuk memastikan kebijakan tersebut mencakup semua cara yang ingin Anda berikan akses ke bucket di luar akun Anda. Setelah menonaktifkan ACL, bucket Anda hanya menerima permintaan PUT yang tidak menentukan ACL atau permintaan PUT dengan ACL kontrol penuh pemilik bucket, seperti ACL `bucket-owner-full-control` terekam atau bentuk setara ACL ini yang dinyatakan dalam XML. Aplikasi yang ada yang mendukung ACL kontrol penuh pemilik bucket tidak akan berdampak. PUTpermintaan yang berisi ACL lain (misalnya, hibah khusus untuk tertentu Akun AWS) gagal dan mengembalikan `400` kesalahan dengan kode kesalahan. `AccessControlListNotSupported`

Sebaliknya, bucket dengan pengaturan pilihan pemilik Bucket terus menerima dan menghormati ACL bucket dan objek. Dengan pengaturan ini, objek baru yang ditulis dengan ACL terekam `bucket-owner-full-control` secara otomatis dimiliki oleh pemilik bucket daripada penulis objek. Semua perilaku ACL lainnya tetap sama. Agar semua operasi PUT Amazon S3 menyertakan ACL `bucket-`

`owner-full-control` yang terekam, Anda dapat menambahkan kebijakan bucket yang hanya mengizinkan unggahan objek menggunakan ACL ini.

Untuk melihat pengaturan Kepemilikan Objek yang diterapkan ke bucket, Anda dapat menggunakan metrik Lensa Penyimpanan Amazon S3. Lensa Penyimpanan S3 adalah fitur analisis penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan objek. Untuk informasi lebih lanjut, lihat Menggunakan Lensa Penyimpanan S3 untuk menemukan pengaturan Kepemilikan Objek.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Pengaturan Kepemilikan Objek

Tabel ini menunjukkan dampak yang dimiliki setiap pengaturan Kepemilikan Objek pada ACL, objek, kepemilikan objek, dan unggahan objek.

Pengaturan	Berlaku untuk	Efek pada kepemilikan objek	Efek pada ACL	Unggahan diterima
Pemilik bucket diberlakukan (default)	Semua objek baru dan yang sudah ada	Pemilik bucket memiliki setiap objek.	ACL dinonaktifkan dan tidak lagi memengaruhi izin akses ke bucket Anda. Permintaan untuk mengatur atau memperbarui ACL gagal. Namun, permintaan untuk membaca ACL didukung.	Unggahan dengan pemilik bucket mengontrol penuh ACL atau unggahan yang tidak menentukan ACL

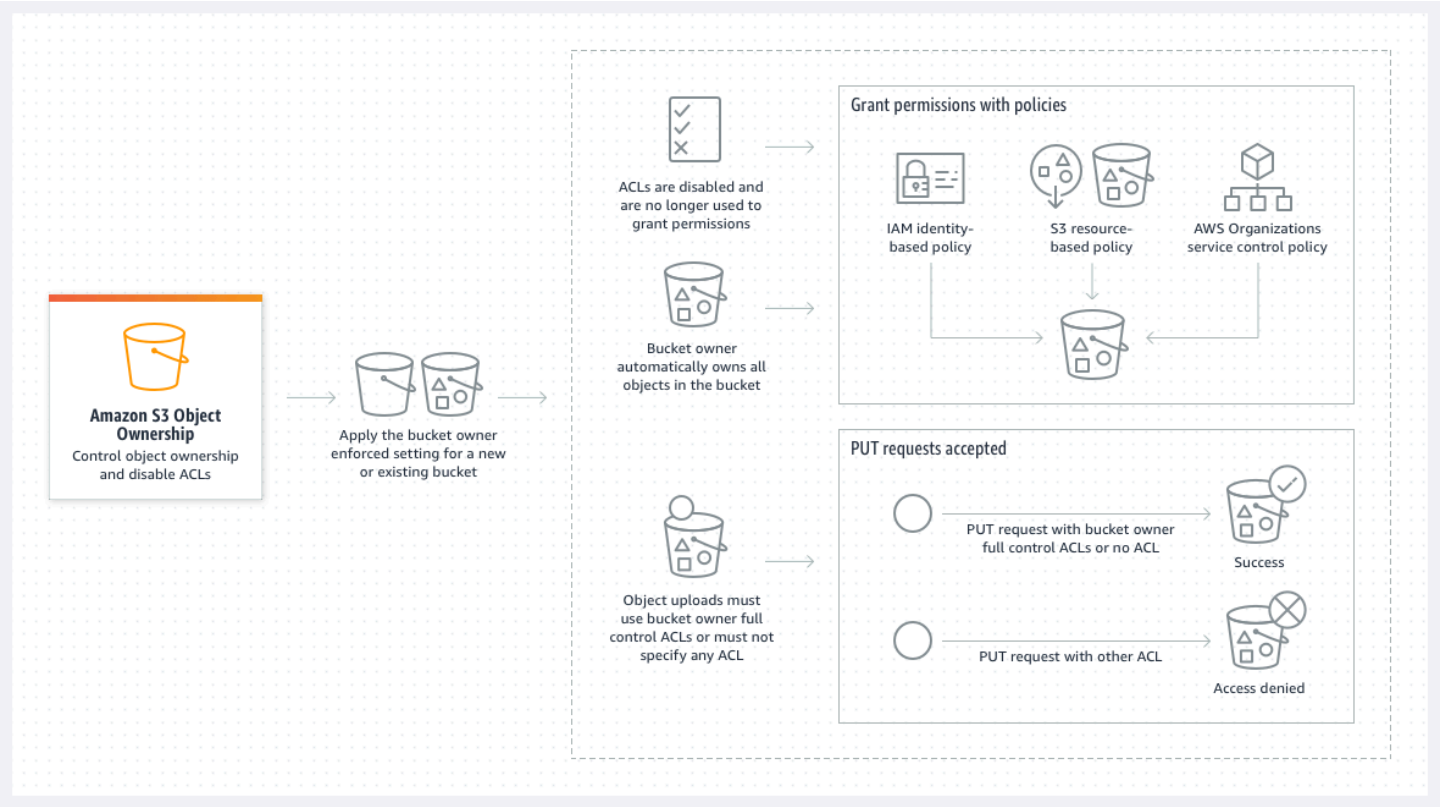
Pengaturan	Berlaku untuk	Efek pada kepemilikan objek	Efek pada ACL	Unggahan diterima
			Pemilik bucket memiliki kepemilikan dan kontrol penuh. Penulis objek tidak lagi memiliki kepemilikan dan kontrol penuh.	
bucket lebih disukai pemilik bucket	Objek baru	Jika unggahan objek menyertakan ACL `bucket-owner-full-control` terekam, pemilik bucket memiliki objek tersebut. Objek yang diunggah dengan ACL lain dimiliki oleh akun penulisan.	ACL dapat diperbaru i dan dapat memberikan izin. Jika unggahan objek menyertakan ACL `bucket-owner-full-control` terekam, pemilik bucket memiliki akses kontrol penuh, dan penulis objek tidak lagi memiliki akses kontrol penuh.	Semua unggahan

Pengaturan	Berlaku untuk	Efek pada kepemilikan objek	Efek pada ACL	Unggahan diterima
Penulis objek	Objek baru	Penulis objek memiliki objek.	ACL dapat diperbarui, dan dapat memberikan izin. Penulis objek memiliki akses kontrol penuh.	Semua unggahan

Perubahan diperkenalkan dengan menonaktifkan ACL

Ketika pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek diterapkan, ACL dinonaktifkan dan Anda secara otomatis memiliki dan mengambil kendali penuh atas setiap objek di bucket tanpa mengambil tindakan tambahan apa pun. Pemilik bucket diberlakukan adalah pengaturan default untuk semua bucket yang baru dibuat. Setelah pengaturan diberlakukan pemilik Bucket diterapkan, Anda akan melihat tiga perubahan:

- Semua ACL bucket dan ACL objek dinonaktifkan, yang memberikan akses penuh kepada Anda, sebagai pemilik bucket. Saat Anda melakukan permintaan ACL baca pada bucket atau objek Anda, Anda akan melihat bahwa akses penuh hanya diberikan kepada pemilik bucket.
- Anda, sebagai pemilik bucket, secara otomatis memiliki dan memiliki kendali penuh atas setiap objek di bucket Anda.
- ACL tidak lagi memengaruhi izin akses ke bucket Anda. Akibatnya, kontrol akses untuk data Anda didasarkan pada kebijakan, seperti kebijakan IAM, kebijakan bucket S3, kebijakan titik akhir VPC, dan Organizations SCP.



Jika Anda menggunakan Pembuatan Penentuan Versi S3, pemilik bucket memiliki dan memiliki kontrol penuh atas semua versi objek di bucket Anda. Menerapkan pengaturan yang diberlakukan pemilik Bucket tidak akan menambahkan versi baru objek.

Objek baru dapat diunggah ke bucket Anda hanya jika mereka menggunakan ACL kontrol penuh pemilik bucket atau tidak menentukan ACL. Unggahan objek gagal jika mereka menentukan ACL lainnya. Untuk informasi selengkapnya, lihat Pemecahan Masalah.

Karena contoh operasi `PutObject` berikut menggunakan AWS Command Line Interface (AWS CLI) menyertakan ACL `bucket-owner-full-control` terekam, objek dapat diunggah ke bucket dengan ACL yang dinonaktifkan.

```
aws s3api put-object --bucket DOC-EXAMPLE-BUCKET --key key-name --body path-to-file --
acl bucket-owner-full-control
```

Karena operasi `PutObject` berikut tidak menentukan ACL, operasi ini juga berhasil untuk bucket dengan ACL yang dinonaktifkan.

```
aws s3api put-object --bucket DOC-EXAMPLE-BUCKET --key key-name --body path-to-file
```

> **Note**
>
> Jika orang lain Akun AWS memerlukan akses ke objek setelah mengunggah, Anda harus memberikan izin tambahan ke akun tersebut melalui kebijakan bucket. Untuk informasi selengkapnya, lihat Panduan yang menggunakan kebijakan untuk mengelola akses ke sumber daya Amazon S3.

Mengaktifkan kembali ACL

Anda dapat mengaktifkan kembali ACL dengan mengubah dari pengaturan diberlakukan pemilik Bucket ke pengaturan Kepemilikan Objek lain kapan saja. Jika Anda menggunakan ACL objek untuk manajemen izin sebelum menerapkan pengaturan diberlakukan pemilik Bucket dan Anda tidak memigrasikan izin ACL objek ini ke kebijakan bucket, setelah Anda mengaktifkan kembali ACL, izin ini akan dipulihkan. Selain itu, objek yang ditulis ke bucket saat pengaturan yang diberlakukan pemilik Bucket diterapkan masih dimiliki oleh pemilik bucket.

Misalnya, jika Anda mengubah dari pengaturan yang diberlakukan pemilik Bucket kembali ke pengaturan Object writer, Anda, sebagai pemilik bucket, tidak lagi memiliki dan memiliki kontrol penuh atas objek yang sebelumnya dimiliki oleh orang lain Akun AWS. Sebaliknya, akun pengunggahan kembali memiliki objek-objek ini. Objek yang dimiliki oleh akun lain menggunakan ACL untuk izin, sehingga Anda tidak dapat menggunakan kebijakan untuk memberikan izin ke objek tersebut. Namun, Anda, sebagai pemilik bucket, masih memiliki objek apa pun yang ditulis ke bucket saat pengaturan yang diberlakukan oleh pemilik Bucket diterapkan. Objek ini tidak dimiliki oleh penulis objek, bahkan jika Anda mengaktifkan kembali ACL.

Untuk petunjuk tentang mengaktifkan dan mengelola ACL menggunakan, (AWS Command Line Interface CLI) AWS Management Console, REST API, atau AWS SDK, lihat. Mengonfigurasi ACL

Prasyarat untuk menonaktifkan ACL

Sebelum Anda menonaktifkan ACL untuk bucket yang ada, selesaikan prasyarat berikut.

Tinjau ACL bucket dan objek dan migrasi izin ACL

Saat Anda menonaktifkan ACL, izin yang diberikan oleh bucket dan ACL objek tidak lagi memengaruhi akses. Sebelum menonaktifkan ACL, tinjau bucket dan ACL objek Anda.

Jika ACL bucket Anda memberikan izin baca atau tulis kepada orang lain di luar akun, Anda harus memigrasikan izin ini ke kebijakan bucket sebelum dapat menerapkan pengaturan yang diberlakukan

pemilik Bucket. Jika Anda tidak memigrasikan ACL bucket yang memberikan akses baca atau tulis di luar akun, permintaan Anda untuk menerapkan pengaturan yang diberlakukan pemilik Bucket gagal dan mengembalikan kode kesalahan InvalidBucketAclWithObjectOwnership.

Misalnya, jika Anda ingin menonaktifkan ACL untuk bucket yang menerima log akses server, Anda harus memigrasikan izin ACL bucket untuk grup pengiriman log S3 ke pengguna utama layanan logging dalam kebijakan bucket. Untuk informasi selengkapnya, lihat Berikan akses ke grup pengiriman log S3 untuk pencatatan akses server.

Jika Anda ingin penulis objek mempertahankan kontrol penuh dari objek yang mereka unggah, penulis objek adalah pengaturan Kepemilikan Objek terbaik untuk kasus penggunaan Anda. Jika Anda ingin mengontrol akses pada tingkat objek individual, pemilik bucket lebih disukai adalah pilihan terbaik. Kasus penggunaan ini jarang terjadi.

Untuk meninjau ACL dan memigrasikan izin ACL ke kebijakan bucket, lihat Prasyarat untuk menonaktifkan ACL.

Identifikasi permintaan yang memerlukan ACL untuk otorisasi

Untuk mengidentifikasi permintaan Amazon S3 yang memerlukan ACL untuk otorisasi, Anda dapat menggunakan nilai `aclRequired` di log akses server Amazon S3, atau AWS CloudTrail. Jika permintaan memerlukan ACL untuk otorisasi atau jika Anda memiliki permintaan PUT yang menentukan ACL, string adalah `Yes`. Jika tidak ada ACL yang diperlukan, atau jika Anda menyetel ACL `bucket-owner-full-control` kalengan, atau jika permintaan diizinkan oleh kebijakan bucket Anda, string `aclRequired` nilainya adalah "-" di log akses server Amazon S3 dan tidak ada di. CloudTrail Untuk informasi selengkapnya tentang nilai `aclRequired` yang diharapkan, lihat Nilai `aclRequired` untuk permintaan Amazon S3 umum.

Jika Anda memiliki permintaan `PutBucketAcl` atau `PutObjectAcl` dengan header yang memberikan izin berbasis ACL, kecuali ACL yang `bucket-owner-full-control` terekam, Anda harus menghapus header tersebut sebelum dapat menonaktifkan ACL. Jika tidak, permintaan Anda akan gagal.

Untuk semua permintaan lain yang memerlukan ACL untuk otorisasi, migrasi izin ACL tersebut ke kebijakan bucket. Kemudian, hapus ACL bucket apa pun sebelum Anda mengaktifkan pengaturan yang diberlakukan pemilik bucket.

> **Note**
>
> Jangan hapus ACL objek. Jika tidak, aplikasi yang mengandalkan ACL objek untuk izin akan kehilangan akses.

Jika Anda melihat bahwa tidak ada permintaan yang memerlukan ACL untuk otorisasi, Anda dapat melanjutkan untuk menonaktifkan ACL. Untuk informasi selengkapnya tentang mengidentifikasi permintaan, lihat Menggunakan log akses server Amazon S3 untuk mengidentifikasi permintaan danMengidentifikasi permintaan Amazon S3 menggunakan CloudTrail.

Meninjau dan memperbarui kebijakan bucket yang menggunakan kunci kondisi terkait ACL

Setelah Anda menerapkan pengaturan diberlakukan pemilik Bucket untuk menonaktifkan ACL, objek baru dapat diunggah ke bucket Anda hanya jika permintaan menggunakan ACL kontrol penuh pemilik bucket atau tidak menentukan ACL. Sebelum menonaktifkan ACL, tinjau kebijakan bucket Anda untuk kunci kondisi terkait ACL.

Jika kebijakan bucket Anda menggunakan kunci kondisi terkait ACL untuk mewajibkan ACL yang terekam `bucket-owner-full-control` (misalnya, `s3:x-amz-acl`), Anda tidak perlu memperbarui kebijakan bucket. Kebijakan bucket berikut menggunakan `s3:x-amz-acl` untuk mewajibkan ACL yang terekam `bucket-owner-full-control` untuk permintaan `PutObject` S3. Kebijakan ini masih mengharuskan penulis objek untuk menentukan ACL terekam `bucket-owner-full-control`. Namun, bucket dengan ACL dinonaktifkan masih menerima ACL ini, sehingga permintaan terus berhasil tanpa perlu perubahan sisi klien.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Only allow writes to my bucket with bucket owner full control",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:user/ExampleUser"
                ]
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
```

```
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control"
                }
            }
        }
    ]
}
```

Namun, jika kebijakan bucket Anda menggunakan kunci kondisi terkait ACL yang memerlukan ACL berbeda, Anda harus menghapus kunci kondisi ini. Contoh kebijakan bucket ini memerlukan `public-read` ACL untuk permintaan S3 `PutObject`, dan karenanya harus diperbarui sebelum menonaktifkan ACL.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Only allow writes to my bucket with public read access",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:user/ExampleUser"               ]
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "public-read"
                }
            }
        }
    ]
}
```

Izin Kepemilikan Objek

Untuk menerapkan, memperbarui, atau menghapus pengaturan Kepemilikan Objek untuk bucket, Anda memerlukan izin `s3:PutBucketOwnershipControls` tersebut. Untuk mengembalikan pengaturan Kepemilikan Objek untuk bucket, Anda memerlukan izin

`s3:GetBucketOwnershipControls`. Untuk informasi selengkapnya, lihat Mengatur Kepemilikan Objek saat membuat bucket dan Melihat pengaturan Kepemilikan Objek untuk bucket S3.

Menonaktifkan ACL untuk semua bucket baru

Secara default, semua bucket baru dibuat dengan pengaturan diberlakukan pemilik Bucket diterapkan dan ACL dinonaktifkan. Kami merekomendasikan agar ACL dinonaktifkan. Sebagai aturan umum, kami menyarankan penggunaan kebijakan berbasis sumber daya S3 (kebijakan bucket dan kebijakan titik akses) atau kebijakan IAM untuk kontrol akses, bukan ACL. Kebijakan adalah opsi kontrol akses yang disederhanakan dan lebih fleksibel. Dengan kebijakan bucket dan kebijakan titik akses, Anda dapat menentukan aturan yang berlaku secara luas di semua permintaan ke sumber daya Amazon S3 Anda.

Replikasi dan Kepemilikan Object

Bila Anda menggunakan replikasi S3 dan bucket sumber dan tujuan dimiliki oleh berbeda Akun AWS, Anda dapat menonaktifkan ACL (dengan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek) untuk mengubah kepemilikan replika ke bucket yang memiliki tujuan. Akun AWS Pengaturan ini meniru perilaku override pemilik yang ada tanpa memerlukan izin `s3:ObjectOwnerOverrideToBucketOwner`. Semua objek yang direplikasi ke bucket tujuan dengan pengaturan yang diberlakukan pemilik Bucket dimiliki oleh pemilik bucket tujuan. Untuk informasi selengkapnya tentang opsi penggantian pemilik untuk konfigurasi replikasi, lihat Mengubah pemilik replika.

Mengatur Kepemilikan Objek

Anda dapat menerapkan setelan Kepemilikan Objek dengan menggunakan konsol Amazon S3, AWS SDK AWS CLI, Amazon S3 REST API, atau. AWS CloudFormation API REST dan AWS CLI perintah berikut mendukung Kepemilikan Objek:

API REST	AWS CLI	Deskripsi
PutBucketOwnershipControls	put-bucket-ownership-controls	Membuat atau memodifikasi pengaturan Kepemilikan Objek untuk bucket S3 yang ada.
CreateBucket	create-bucket	Membuat bucket menggunakan header permintaan `x-amz-object-ownership`

API REST	AWS CLI	Deskripsi
		untuk menentukan pengaturan Kepemilikan Objek.
GetBucketOwnershipControls	get-bucket-ownership-controls	Mengambil pengaturan Kepemilikan Objek untuk bucket Amazon S3.
DeleteBucketOwnershipControls	delete-bucket-ownership-controls	Menghapus pengaturan Kepemilikan Objek untuk bucket Amazon S3.

Untuk informasi tentang menerapkan dan bekerja dengan pengaturan Kepemilikan Objek, lihat topik-topik berikut.

Topik

- Prasyarat untuk menonaktifkan ACL
- Mengatur Kepemilikan Objek saat membuat bucket
- Menyetel Kepemilikan Objek pada bucket yang ada
- Melihat pengaturan Kepemilikan Objek untuk bucket S3
- Menonaktifkan ACL untuk semua bucket baru dan menegakkan Kepemilikan Objek
- Pemecahan Masalah

Prasyarat untuk menonaktifkan ACL

Jika ACL bucket Anda memberikan akses di luar Akun AWS, sebelum menonaktifkan ACL, Anda harus memigrasikan izin ACL bucket ke kebijakan bucket dan mengatur ulang ACL bucket Anda ke ACL pribadi default. Jika Anda tidak memigrasikan ACL bucket ini, permintaan Anda untuk menerapkan pengaturan yang diberlakukan pemilik Bucket untuk menonaktifkan ACL gagal dan mengembalikan kode kesalahan InvalidBucketAclWithObjectOwnership. Kami juga menyarankan Anda meninjau izin ACL objek dan memigrasikannya ke kebijakan bucket Anda. Untuk informasi lebih lanjut tentang prasyarat lain yang disarankan, lihat Prasyarat untuk menonaktifkan ACL.

Setiap bucket dan ACL objek yang ada memiliki kesetaraan dalam kebijakan IAM. Contoh kebijakan bucket berikut menunjukkan cara `READ` dan izin `WRITE` untuk bucket dan objek ACL memetakan ke

izin IAM. Untuk informasi selengkapnya tentang bagaimana setiap ACL menerjemahkan ke izin IAM, lihat Pemetaan izin ACL dan izin kebijakan akses.

Untuk meninjau dan memigrasi izin ACL ke kebijakan bucket, lihat topik berikut.

Topik

- Contoh kebijakan bucket
- Menggunakan konsol S3 untuk meninjau dan memigrasikan izin ACL
- Menggunakan AWS CLI untuk meninjau dan memigrasi izin ACL
- Contoh panduan

Contoh kebijakan bucket

Contoh kebijakan bucket ini menunjukkan cara memigrasi `READ` dan `WRITE` bucket serta objek izin ACL untuk Akun AWS pihak ketiga ke kebijakan bucket. ACL `READ_ACP`dan `WRITE_ACP` kurang relevan untuk kebijakan tersebut, sebab mereka memberikan izin terkait ACL (`s3:GetBucketAcl`, `s3:GetObjectAcl`, `s3:PutBucketAcl`, dan `s3:PutObjectAcl`).

Example – **`READ`** ACL untuk bucket

Jika bucket Anda memiliki `READ` ACL yang memberikan Akun AWS *111122223333* izin untuk mencantumkan konten bucket, Anda dapat menulis kebijakan bucket yang memberikan `s3:ListBucket` `s3:ListBucketMultipartUploads` izin untuk bucket Anda. `s3:ListBucketVersions`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Permission to list the objects in a bucket",
            "Effect": "Allow",
            "Principal": {
                "AWS": [

                    "arn:aws:iam::111122223333:root"
                ]
            },
            "Action": [
                "s3:ListBucket",
                "s3:ListBucketVersions",
```

```
                "s3:ListBucketMultipartUploads"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
        }
    ]
}
```

Example — ACL **READ** untuk setiap objek dalam bucket

Jika setiap objek di bucket Anda memiliki `READ` ACL yang memberikan akses ke Akun AWS *111122223333*, Anda dapat menulis kebijakan bucket yang memberikan `s3:GetObject` dan `s3:GetObjectVersion` izin ke akun ini untuk setiap objek di bucket Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Read permission for every object in a bucket",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root"
                ]
            },
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        }
    ]
}
```

Contoh elemen sumber daya ini memberikan akses ke objek tertentu.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/OBJECT-KEY"
```

Example – ACL **WRITE** yang memberikan izin untuk menulis objek ke bucket

Jika bucket Anda memiliki `WRITE` ACL yang memberikan Akun AWS *111122223333* izin untuk menulis objek ke bucket, Anda dapat menulis kebijakan bucket yang memberikan `s3:PutObject` izin untuk bucket Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Permission to write objects to a bucket",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root"
                ]
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        }
    ]
}
```

Menggunakan konsol S3 untuk meninjau dan memigrasikan izin ACL

Untuk meninjau izin ACL bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Bucket, pilih nama bucket.
3. Pilih tab Izin.
4. Di bagian bawah Daftar kontrol akses (ACL), tinjau izin ACL bucket Anda.

Untuk meninjau izin ACL objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang berisi objek Anda.
3. Di daftar Objek, pilih nama objek Anda.
4. Pilih tab Izin.
5. Di bagian bawah Daftar kontrol akses (ACL), tinjau izin ACL objek Anda.

Untuk memigrasi izin ACL dan memperbarui bucket ACL

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Bucket, pilih nama bucket.
3. Pada tab Izin, di bawah Kebijakan bucket, pilih Edit.
4. Di kotak Kebijakan, tambahkan atau perbarui kebijakan bucket Anda.

   Misalnya kebijakan bucket, lihat Contoh kebijakan bucket danContoh panduan.
5. Pilih Simpan perubahan.
6. Perbarui bucket ACL Anda untuk menghapus pemberian ACL ke grup lain atau. Akun AWS
7. Terapkan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek.

Menggunakan AWS CLI untuk meninjau dan memigrasi izin ACL

1. Untuk mengembalikan bucket ACL untuk bucket Anda, gunakan get-bucket-acl AWS CLI perintah:

   ```
   aws s3api get-bucket-acl --bucket DOC-EXAMPLE-BUCKET
   ```

   Misalnya, bucket ACL ini memberikan `WRITE` dan `READ` akses ke akun pihak ketiga. Dalam ACL ini, akun pihak ketiga diidentifikasi oleh ID pengguna kanonik. Untuk menerapkan pengaturan yang diberlakukan pemilik Bucket dan menonaktifkan ACL, Anda harus memigrasikan izin ini untuk akun pihak ketiga ke kebijakan bucket.

   ```
   {
       "Owner": {
           "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
           "ID": "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID"
       },
       "Grants": [
           {
               "Grantee": {
                   "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
                   "ID":
    "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID",
                   "Type": "CanonicalUser"
               },
   ```

```
            "Permission": "FULL_CONTROL"
        },
        {
            "Grantee": {
                "DisplayName": "THIRD-PARTY-EXAMPLE-ACCOUNT",
                "ID":
 "72806de9d1ae8b171cca9e2494a8d1335dfced4ThirdPartyAccountCanonicalUserID",
                "Type": "CanonicalUser"
            },
            "Permission": "READ"
        },
        {
            "Grantee": {
                "DisplayName": "THIRD-PARTY-EXAMPLE-ACCOUNT",
                "ID":
 "72806de9d1ae8b171cca9e2494a8d1335dfced4ThirdPartyAccountCanonicalUserID",
                "Type": "CanonicalUser"
            },
            "Permission": "WRITE"
        }
    ]
}
```

Untuk contoh ACL lainnya, lihat Contoh panduan.

2. Migrasikan izin ACL bucket Anda ke kebijakan bucket:

   Contoh kebijakan bucket ini memberikan izin `s3:PutObject` dan `s3:ListBucket` untuk akun pihak ketiga. Dalam kebijakan bucket, akun pihak ketiga diidentifikasi oleh Akun AWS ID (*111122223333*).

```
aws s3api put-bucket-policy --bucket DOC-EXAMPLE-BUCKET --policy file://policy.json

policy.json:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PolicyForCrossAccountAllowUpload",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:root"
```

```
                ]
            },
            "Action": [
                "s3:PutObject",
                "s3:ListBucket"
            ],
            "Resource": [
             "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
             "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        }
    ]
}
```

Untuk contoh kebijakan bucket lainnya, lihat Contoh kebijakan bucket dan Contoh panduan.

3. Untuk mengembalikan ACL untuk objek tertentu, gunakan get-object-acl AWS CLI perintah.

```
aws s3api get-object-acl --bucket DOC-EXAMPLE-BUCKET --key EXAMPLE-OBJECT-KEY
```

4. Jika diperlukan, migrasi izin ACL objek ke kebijakan bucket Anda.

   Contoh elemen sumber daya ini memberikan akses ke objek tertentu dalam kebijakan bucket.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/EXAMPLE-OBJECT-KEY"
```

5. Setel ulang ACL untuk bucket Anda ke ACL default.

```
aws s3api put-bucket-acl --bucket DOC-EXAMPLE-BUCKET --acl private
```

6. Terapkan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek.

Contoh panduan

Contoh berikut ini menunjukkan cara memigrasikan izin ACL ke kebijakan bucket untuk kasus penggunaan tertentu.

Topik

- Berikan akses ke grup pengiriman log S3 untuk pencatatan akses server
- Berikan akses baca publik ke objek dalam bucket
- Berikan Amazon ElastiCache untuk Redis akses ke bucket S3 Anda

Berikan akses ke grup pengiriman log S3 untuk pencatatan akses server

Jika Anda ingin menerapkan pengaturan yang diberlakukan pemilik Bucket untuk menonaktifkan ACL untuk bucket tujuan pencatatan akses server (juga dikenal sebagai bucket target), Anda harus memigrasikan izin ACL bucket untuk grup pengiriman log S3 ke layanan logging pengguna utama (`logging.s3.amazonaws.com`) dalam kebijakan bucket. Untuk informasi lebih lanjut tentang izin pengiriman log, lihat Izin untuk pengiriman log.

ACL bucket ini memberikan akses `WRITE` dan `READ_ACP` ke grup pengiriman log S3:

```
{
    "Owner": {
        "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
        "ID": "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID"
    },
    "Grants": [
        {
            "Grantee": {
                "Type": "CanonicalUser",
                "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
                "ID":
 "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID"
            },
            "Permission": "FULL_CONTROL"
        },
        {
            "Grantee": {
                "Type": "Group",
                "URI": "http://acs.amazonaws.com/groups/s3/LogDelivery"
            },
            "Permission": "WRITE"
        },
        {
            "Grantee": {
                "Type": "Group",
                "URI": "http://acs.amazonaws.com/groups/s3/LogDelivery"
            },
            "Permission": "READ_ACP"
        }
    ]
}
```

Untuk memigrasikan izin ACL bucket untuk grup pengiriman log S3 ke pengguna utama layanan logging dalam kebijakan bucket

1. Tambahkan kebijakan bucket berikut ke bucket tujuan Anda, ganti nilai contoh.

```
aws s3api put-bucket-policy --bucket DOC-EXAMPLE-BUCKET --policy file://policy.json

policy.json:      {
    {
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ServerAccessLogsPolicy",
            "Effect": "Allow",
            "Principal": {
                "Service": "logging.s3.amazonaws.com"
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/EXAMPLE-LOGGING-PREFIX*",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:::SOURCE-BUCKET-NAME"
                },
                "StringEquals": {
                    "aws:SourceAccount": "SOURCE-AWS-ACCOUNT-ID"
                }
            }
        }
    ]
}
```

2. Setel ulang ACL untuk bucket tujuan Anda ke ACL default.

```
aws s3api put-bucket-acl --bucket DOC-EXAMPLE-BUCKET --acl private
```

3. Terapkan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek ke bucket tujuan Anda.

Berikan akses baca publik ke objek dalam bucket

Jika ACL objek Anda memberikan akses baca publik ke semua objek di bucket, Anda dapat memigrasikan izin ACL ini ke kebijakan bucket.

Objek ACL ini memberikan akses baca publik ke objek dalam bucket:

```
{
    "Owner": {
        "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
        "ID": "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
                "ID":
 "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID",
                "Type": "CanonicalUser"
            },
            "Permission": "FULL_CONTROL"
        },
        {
            "Grantee": {
                "Type": "Group",
                "URI": "http://acs.amazonaws.com/groups/global/AllUsers"
            },
            "Permission": "READ"
        }
    ]
}
```

Untuk memigrasikan izin ACL baca publik ke kebijakan bucket

1. Untuk memberikan akses baca publik ke semua objek di bucket Anda, tambahkan kebijakan bucket berikut, ganti nilai contoh.

```
aws s3api put-bucket-policy --bucket DOC-EXAMPLE-BUCKET --policy file://policy.json

policy.json:
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        }
    ]
}
```

Untuk memberikan akses publik ke objek tertentu dalam kebijakan bucket, gunakan format berikut untuk elemen `Resource` tersebut.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/OBJECT-KEY"
```

Untuk memberikan akses publik ke semua objek dengan prefiks tertentu, gunakan format berikut untuk elemen `Resource` tersebut.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/PREFIX/*"
```

2. Terapkan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek.

Berikan Amazon ElastiCache untuk Redis akses ke bucket S3 Anda

Anda dapat mengekspor ElastiCache cadangan Redis Anda ke ember S3, yang memberi Anda akses ke cadangan dari luar. ElastiCache Untuk mengekspor cadangan ke bucket S3, Anda harus memberikan ElastiCache izin untuk menyalin snapshot ke bucket. Jika Anda telah memberikan izin ke ElastiCache ACL bucket, Anda harus memigrasikan izin ini ke kebijakan bucket sebelum menerapkan pengaturan yang diberlakukan pemilik Bucket untuk menonaktifkan ACL. Untuk informasi selengkapnya, lihat Memberikan ElastiCache akses ke bucket Amazon S3 Anda di ElastiCache Panduan Pengguna Amazon.

Contoh berikut menunjukkan izin ACL bucket yang memberikan izin. ElastiCache

```
{
    "Owner": {
```

```
        "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
        "ID": "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "DOC-EXAMPLE-ACCOUNT-OWNER",
                "ID":
 "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID",
                "Type": "CanonicalUser"
            },
            "Permission": "FULL_CONTROL"
        },
        {
            "Grantee": {
                "DisplayName": "aws-scs-s3-readonly",
                "ID":
 "540804c33a284a299d2547575ce1010f2312ef3da9b3a053c8bc45bf233e4353",
                "Type": "CanonicalUser"
            },
            "Permission": "READ"
        },
        {
            "Grantee": {
                "DisplayName": "aws-scs-s3-readonly",
                "ID":
 "540804c33a284a299d2547575ce1010f2312ef3da9b3a053c8bc45bf233e4353",
                "Type": "CanonicalUser"
            },
            "Permission": "WRITE"
        },
        {
            "Grantee": {
                "DisplayName": "aws-scs-s3-readonly",
                "ID":
 "540804c33a284a299d2547575ce1010f2312ef3da9b3a053c8bc45bf233e4353",
                "Type": "CanonicalUser"
            },
            "Permission": "READ_ACP"
        }
    ]
}
```

Untuk memigrasikan izin ACL bucket ElastiCache untuk Redis ke kebijakan bucket

1. Tambahkan kebijakan bucket berikut ke bucket Anda, ganti nilai contoh.

```
aws s3api put-bucket-policy --bucket DOC-EXAMPLE-BUCKET --policy file://policy.json

policy.json:
"Id": "Policy15397346",
    "Statement": [
        {
            "Sid": "Stmt15399483",
            "Effect": "Allow",
            "Principal": {
                "Service": "Region.elasticache-snapshot.amazonaws.com"
            },
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:GetBucketAcl",
                "s3:ListMultipartUploadParts",
                "s3:ListBucketMultipartUploads"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        }
    ]
}
```

2. Setel ulang ACL untuk bucket Anda ke ACL default:

```
aws s3api put-bucket-acl --bucket DOC-EXAMPLE-BUCKET --acl private
```

3. Terapkan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek.

Mengatur Kepemilikan Objek saat membuat bucket

Saat membuat bucket, Anda dapat mengonfigurasi Kepemilikan S3 Object. Untuk mengatur Kepemilikan Objek untuk bucket yang ada, lihat Menyetel Kepemilikan Objek pada bucket yang ada.

Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk menonaktifkan daftar kontrol akses (ACL) dan mengambil kepemilikan setiap objek di bucket Anda, menyederhanakan manajemen akses untuk data yang disimpan di Amazon S3. Secara default, Kepemilikan Objek S3 diatur ke pengaturan yang diberlakukan pemilik Bucket, dan ACL dinonaktifkan untuk bucket baru. Dengan menonaktifkan ACL, pemilik bucket memiliki setiap objek di bucket dan mengelola akses ke data secara eksklusif menggunakan kebijakan manajemen akses. Kami menyarankan agar ACL dinonaktifkan, kecuali dalam keadaan yang tidak biasa ketika Anda perlu mengontrol akses untuk setiap objek secara individual.

Kepemilikan Objek memiliki tiga pengaturan yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda, dan untuk menonaktifkan atau mengaktifkan ACL:

ACL dinonaktifkan

- Pemilik bucket yang diberlakukan (default)–ACL dinonaktifkan, dan pemilik bucket secara otomatis memilikinya, serta mendapatkan kontrol penuh atas setiap objek di dalam bucket. ACL tidak lagi memengaruhi izin terhadap data di bucket S3. Bucket tersebut menggunakan kebijakan untuk menentukan kontrol akses.

ACL diaktifkan

- Pemilik bucket yang dipilih—Pemilik bucket memiliki dan diberikan kendali penuh atas objek baru yang ditulis akun lain ke bucket dengan ACL `bucket-owner-full-control` yang dibatasi.
- Penulis objek — Akun AWS Yang mengunggah objek memiliki objek, memiliki kontrol penuh atasnya, dan dapat memberikan pengguna lain akses ke sana melalui ACL.

Izin: Untuk menerapkan pengaturan yang diberlakukan pemilik Bucket atau pengaturan pilihan pemilik Bucket, Anda harus memiliki izin berikut: `s3:CreateBucket` dan `s3:PutBucketOwnershipControls`. Tidak diperlukan izin tambahan saat membuat bucket dengan pengaturan Object writer yang diterapkan. Untuk informasi selengkapnya tentang izin Amazon S3, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

> **Important**
>
> Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL, dan kami menyarankan Anda menonaktifkan ACL kecuali dalam keadaan yang tidak biasa di mana Anda perlu mengontrol akses untuk setiap objek satu per satu. Dengan

Kepemilikan Objek, Anda dapat menonaktifkan ACL dan mengandalkan kebijakan untuk kontrol akses. Saat menonaktifkan ACL, Anda dapat dengan mudah memelihara bucket berisi objek yang diunggah oleh akun yang berbeda AWS . Anda, sebagai pemilik bucket, memiliki semua objek di bucket dan dapat mengelola akses ke mereka menggunakan kebijakan.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di bilah navigasi di bagian atas halaman, pilih nama yang saat ini ditampilkan Wilayah AWS. Selanjutnya, pilih Wilayah tempat Anda ingin membuat ember.

   > **Note**
   >
   > Untuk meminimalkan latensi dan biaya serta memenuhi persyaratan regulasi, pilih Wilayah yang dekat dengan Anda. Objek yang disimpan di Wilayah tidak pernah keluar dari Wilayah kecuali Anda secara tegas mentransfer atau mereplikasinya ke Wilayah lain. Untuk daftar Amazon S3 Wilayah AWS, lihat Layanan AWS titik akhir di. Referensi Umum Amazon Web Services

3. Di panel navigasi kiri, pilih Bucket.
4. Pilih Buat bucket.

   Halaman Buat bucket terbuka.

5. Di bawah Konfigurasi umum, lihat Wilayah AWS tempat bucket Anda akan dibuat.
6. Di bawah jenis Bucket, pilih Tujuan umum.
7. Untuk Nama bucket, masukkan nama untuk bucket Anda.

   Nama bucket harus:

   - Unik dalam partisi. Partisi adalah pembuatan grup Wilayah. Saat ini, AWS memiliki tiga partisi: `aws` (Wilayah Standar), `aws-cn` (Wilayah Tiongkok), dan `aws-us-gov` (AWS GovCloud (US) Regions).
   - Panjangnya antara 3 hingga 63 karakter.

- Hanya terdiri dari huruf kecil, angka, titik (.), dan tanda hubung (-). Untuk kompatibilitas terbaik, kami menyarankan agar Anda menghindari penggunaan titik (.) dalam nama bucket, kecuali untuk bucket yang digunakan hanya untuk menghosting situs web statis.
- Dimulai dan diakhiri dengan huruf atau angka.

Setelah membuat bucket, Anda tidak dapat mengubah namanya. Untuk informasi selengkapnya tentang penamaan bucket, lihat Peraturan penamaan bucket.

> **Important**
>
> Hindari menyertakan informasi sensitif, seperti nomor akun, dalam nama bucket. Nama bucket terlihat dalam URL yang menunjuk objek dalam bucket.

8. AWS Management Console memungkinkan Anda menyalin pengaturan bucket yang ada ke bucket baru Anda. Jika Anda tidak ingin menyalin pengaturan bucket yang ada, lewati ke langkah berikutnya.

> **Note**
>
> Opsi ini:
>
> - Tidak tersedia di AWS CLI dan hanya tersedia di konsol
> - Tidak tersedia untuk bucket direktori
> - Tidak menyalin kebijakan bucket dari bucket yang ada ke bucket baru

Untuk menyalin setelan bucket yang ada, di bagian Salin setelan dari bucket yang ada, pilih Pilih bucket. Jendela Select bucket terbuka. Temukan bucket dengan pengaturan yang ingin Anda salin, lalu pilih Pilih bucket. Jendela Choose bucket ditutup, dan jendela Create bucket terbuka kembali.

Di bawah Salin pengaturan dari bucket yang ada, Anda sekarang akan melihat nama bucket yang Anda pilih. Anda juga akan melihat opsi Restore default yang dapat Anda gunakan untuk menghapus pengaturan bucket yang disalin. Tinjau setelan bucket yang tersisa, di halaman Buat bucket. Anda akan melihat bahwa mereka sekarang cocok dengan pengaturan ember yang Anda pilih. Anda dapat melompat ke langkah terakhir.

9. Di bawah Kepemilikan Objek, untuk menonaktifkan atau mengaktifkan ACL dan mengontrol kepemilikan objek yang diunggah di bucket Anda, pilih salah satu pengaturan berikut ini:

   ACL dinonaktifkan

   - Pemilik bucket yang diberlakukan (default)–ACL dinonaktifkan, dan pemilik bucket secara otomatis memilikinya, serta mendapatkan kontrol penuh atas setiap objek di dalam bucket. ACL tidak lagi memengaruhi izin akses ke data di bucket S3. Bucket menggunakan kebijakan secara eksklusif untuk menentukan kontrol akses.

     Secara default, ACL dinonaktifkan. Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL. Kami menyarankan agar ACL dinonaktifkan, kecuali dalam keadaan yang tidak biasa ketika Anda perlu mengontrol akses untuk setiap objek secara individual. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

   ACL diaktifkan

   - Pemilik bucket yang dipilih—Pemilik bucket memiliki dan diberikan kendali penuh atas objek baru yang ditulis akun lain ke bucket dengan ACL `bucket-owner-full-control` yang dibatasi.

     Jika Anda menerapkan pengaturan Pemilik bucket yang dipilih, agar semua unggahan Amazon S3 menyertakan ACL `bucket-owner-full-control` yang terekam, Anda dapat menambahkan kebijakan bucket yang hanya mengizinkan unggahan objek yang menggunakan ACL ini.
   - Penulis objek — Akun AWS Yang mengunggah objek memiliki objek, memiliki kontrol penuh atasnya, dan dapat memberikan pengguna lain akses ke sana melalui ACL.

> **Note**
>
> Pengaturan default-nya adalah Pemilik Bucket yang diberlakukan. Untuk menerapkan pengaturan default dan menonaktifkan ACL, hanya izin `s3:CreateBucket` yang diperlukan. Untuk mengaktifkan ACL, Anda harus memiliki izin `s3:PutBucketOwnershipControls`.

10. Di bawah Pengaturan Blokir Akses Publik untuk bucket ini, pilih pengaturan Blokir Akses Publik yang ingin Anda terapkan ke bucket.

    Secara default, semua pengaturan Blokir Akses Publik untuk bucket direktori diaktifkan. Kami menyarankan Anda tetap mengaktifkan semua pengaturan, kecuali Anda tahu bahwa Anda perlu menonaktifkan satu atau beberapa pengaturan untuk kasus penggunaan spesifik Anda. Untuk informasi lebih lanjut tentang pemblokiran akses publik, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

    > **Note**
    >
    > Untuk mengaktifkan semua pengaturan Blokir Akses Publik, hanya izin `s3:CreateBucket` yang diperlukan. Untuk mematikan pengaturan Blokir Akses Publik, Anda harus memiliki izin `s3:PutBucketPublicAccessBlock`.

11. (Opsional) Di bawah Penentuan Versi Bucket, Anda dapat memilih apakah Anda ingin menyimpan varian objek di bucket Anda. Untuk informasi selengkapnya tentang penentuan versi, lihat Menggunakan Penentuan Versi dalam bucket S3.

    Untuk menonaktifkan atau mengaktifkan Penentuan Versi di bucket Anda, pilih Nonaktifkan atau Aktifkan.

12. (Opsional) Di bawah Tanda, Anda dapat memilih untuk menambahkan tanda ke bucket Anda. Tanda adalah pasangan kunci-nilai yang digunakan untuk mengategorikan penyimpanan.

    Untuk menambahkan tanda bucket, masukkan Kunci dan secara opsional Nilai, lalu pilih Tambahkan Tanda.

13. Di bagian bawah Enkripsi default, pilih Edit.

14. Untuk mengonfigurasi enkripsi default, di bawah Jenis enkripsi, pilih salah satu dari berikut ini:

    - Kunci yang dikelola Amazon S3 (SSE-S3)
    - AWS Key Management Service kunci (SSE-KMS)

    > **Important**
    >
    > Jika Anda menggunakan opsi SSE-KMS untuk konfigurasi enkripsi default, Anda tunduk pada kuota permintaan per detik (RPS) AWS KMS. Untuk informasi

selengkapnya tentang AWS KMS kuota dan cara meminta kenaikan kuota, lihat Kuota di Panduan Pengembang AWS Key Management Service .

Bucket dan objek baru dienkripsi dengan enkripsi di sisi server dengan kunci yang dikelola Amazon S3 sebagai tingkat dasar konfigurasi enkripsi. Untuk informasi selengkapnya tentang enkripsi default, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3.

Untuk informasi selengkapnya tentang penggunaan enkripsi di sisi server Amazon S3 guna mengenkripsi data Anda, lihat Menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3).

15. Jika Anda memilih kunci AWS Key Management Service (SSE-KMS), lakukan hal berikut:

    a. Di bawah AWS KMS kunci, tentukan kunci KMS Anda dengan salah satu cara berikut ini:

    - Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari Anda AWS KMS keys, dan pilih kunci KMS Anda dari daftar kunci yang tersedia.

      Kunci Kunci yang dikelola AWS (`aws/s3`) dan kunci terkelola pelanggan Anda muncul dalam daftar ini. Untuk informasi selengkapnya tentang CMK, lihat Kunci pelanggan dan AWS kunci di AWS Key Management Service Panduan Pengembang.
    - Untuk memasukkan ARN kunci KMS, pilih Masukkan AWS KMS key ARN, dan masukkan ARN kunci KMS Anda di bidang yang muncul.
    - Untuk membuat kunci terkelola pelanggan baru di AWS KMS konsol, pilih Buat kunci KMS.

      Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat kunci di Panduan AWS Key Management Service Pengembang.

    > **Important**
    >
    > Anda hanya dapat menggunakan tombol KMS yang tersedia Wilayah AWS sama dengan bucket. Konsol Amazon S3 hanya mencantumkan kunci 100 KMS pertama di Wilayah yang sama dengan bucket. Untuk menggunakan kunci KMS yang tidak terdaftar, Anda harus memasukkan ARN kunci KMS Anda. Jika Anda ingin menggunakan kunci KMS yang dimiliki oleh akun lain, Anda harus terlebih dahulu memiliki izin untuk menggunakan kunci tersebut, dan kemudian Anda harus

memasukkan ARN kunci KMS. Untuk informasi selengkapnya tentang izin lintas akun untuk kunci KMS, lihat Membuat kunci KMS yang dapat digunakan akun lain di Panduan Pengembang AWS Key Management Service . Untuk informasi selengkapnya tentang SSE-KMS, lihat Menentukan enkripsi di sisi server dengan AWS KMS (SSE-KMS).

Saat Anda menggunakan enkripsi sisi server AWS KMS key untuk Amazon S3, Anda harus memilih kunci KMS enkripsi simetris. Amazon S3 hanya mendukung kunci KMS enkripsi simetris dan tidak mendukung kunci KMS asimetris. Untuk informasi selengkapnya, lihat Mengidentifikasi tombol KMS simetris dan asimetris dalam Panduan Pengembang AWS Key Management Service .

Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat kunci di Panduan AWS Key Management Service Pengembang. Untuk informasi selengkapnya tentang penggunaan AWS KMS dengan Amazon S3, lihat. Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS)

b. Saat mengonfigurasi bucket untuk menggunakan enkripsi default dengan SSE-KMS, Anda juga dapat mengaktifkan Kunci Bucket S3. S3 Bucket Keys menurunkan biaya enkripsi dengan mengurangi lalu lintas permintaan dari Amazon S3 ke. AWS KMS Untuk informasi selengkapnya, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3.

   Untuk menggunakan Kunci Bucket S3, di bagian bawah Kunci Bucket, pilih Aktifkan.

16. (Opsional) Jika Anda ingin mengaktifkan Kunci Objek S3, lakukan hal berikut ini:

    a. Pilih Pengaturan lanjutan.

    > **Important**
    >
    > Mengaktifkan Kunci Objek juga mengaktifkan Penentuan Versi untuk bucket. Setelah mengaktifkan, Anda harus mengonfigurasi pengaturan penyimpanan default Kunci Objek dan penahanan legal untuk melindungi objek baru agar tidak dihapus atau ditimpa.

    b. Jika Anda ingin mengaktifkan Kunci Objek, pilih Aktifkan, baca peringatan yang muncul, lalu setujui.

Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek S3.

> **Note**
>
> Untuk membuat bucket dengan dukungan Kunci Objek, Anda harus memiliki izin berikut: `s3:CreateBucket`, `s3:PutBucketVersioning` dan `s3:PutBucketObjectLockConfiguration`.

17. Pilih Buat bucket.

Menggunakan AWS CLI

Untuk menyetel Object Ownership saat membuat bucket baru, gunakan `create-bucket` AWS CLI perintah dengan `--object-ownership` parameter.

Contoh ini menerapkan pengaturan yang diberlakukan pemilik Bucket untuk bucket baru menggunakan: AWS CLI

```
aws s3api create-bucket --bucket  DOC-EXAMPLE-BUCKET --region us-east-1 --object-
ownership BucketOwnerEnforced
```

> **Important**
>
> Jika Anda tidak menyetel Kepemilikan Objek saat membuat bucket menggunakan AWS CLI, pengaturan defaultnya adalah `ObjectWriter` (ACL diaktifkan).

Menggunakan AWS SDK for Java

Contoh ini menyetel pengaturan yang diberlakukan pemilik Bucket untuk bucket baru menggunakan: AWS SDK for Java

```
    // Build the ObjectOwnership for CreateBucket
    CreateBucketRequest createBucketRequest = CreateBucketRequest.builder()
            .bucket(bucketName)
            .objectOwnership(ObjectOwnership.BucketOwnerEnforced)
            .build()

     // Send the request to Amazon S3
     s3client.createBucket(createBucketRequest);
```

Menggunakan AWS CloudFormation

Untuk menggunakan `AWS::S3::Bucket` AWS CloudFormation sumber daya untuk menyetel Kepemilikan Objek saat membuat bucket baru, lihat AWS::S3::Bucket di OwnershipControlsdalam Panduan AWS CloudFormation Pengguna.

Penggunaan API REST

Untuk menerapkan pengaturan diberlakukan pemilik Bucket untuk Kepemilikan S3 Object, gunakan operasi `CreateBucket` API dengan header `x-amz-object-ownership` permintaan disetel ke `BucketOwnerEnforced`. Untuk informasi dan contoh, lihat CreateBucketReferensi API Amazon Simple Storage Service.

Langkah selanjutnya: Setelah menerapkan pengaturan pilihan pemilik Bucket yang diberlakukan atau pemilik bucket untuk Kepemilikan Objek, Anda dapat mengambil langkah-langkah berikut:

- Pemilik bucket yang diberlakukan–Mengharuskan semua bucket baru dibuat dengan ACL dinonaktifkan menggunakan kebijakan IAM atau Organisasi.
- Pemilik bucket yang dipilih–Menambahkan kebijakan bucket S3 untuk ACL terekam `bucket-owner-full-control` untuk semua unggahan objek ke bucket Anda.

Menyetel Kepemilikan Objek pada bucket yang ada

Anda dapat mengonfigurasi Kepemilikan S3 Object pada bucket S3 yang ada. Untuk menerapkan Kepemilikan Objek saat membuat bucket, lihat Mengatur Kepemilikan Objek saat membuat bucket.

Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk menonaktifkan daftar kontrol akses (ACL) dan mengambil kepemilikan setiap objek di bucket Anda, menyederhanakan manajemen akses untuk data yang disimpan di Amazon S3. Secara default, Kepemilikan Objek S3 diatur ke pengaturan yang diberlakukan pemilik Bucket, dan ACL dinonaktifkan untuk bucket baru. Dengan menonaktifkan ACL, pemilik bucket memiliki setiap objek di bucket dan mengelola akses ke data secara eksklusif menggunakan kebijakan manajemen akses. Kami menyarankan agar ACL dinonaktifkan, kecuali dalam keadaan yang tidak biasa ketika Anda perlu mengontrol akses untuk setiap objek secara individual.

Kepemilikan Objek memiliki tiga pengaturan yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda, dan untuk menonaktifkan atau mengaktifkan ACL:

ACL dinonaktifkan

- Pemilik bucket yang diberlakukan (default)–ACL dinonaktifkan, dan pemilik bucket secara otomatis memilikinya, serta mendapatkan kontrol penuh atas setiap objek di dalam bucket. ACL tidak lagi memengaruhi izin terhadap data di bucket S3. Bucket tersebut menggunakan kebijakan untuk menentukan kontrol akses.

ACL diaktifkan

- Pemilik bucket yang dipilih—Pemilik bucket memiliki dan diberikan kendali penuh atas objek baru yang ditulis akun lain ke bucket dengan ACL `bucket-owner-full-control` yang dibatasi.
- Penulis objek — Akun AWS Yang mengunggah objek memiliki objek, memiliki kontrol penuh atasnya, dan dapat memberikan pengguna lain akses ke sana melalui ACL.

Prasyarat: Sebelum menerapkan pengaturan yang diberlakukan pemilik Bucket untuk menonaktifkan ACL, Anda harus memigrasikan izin ACL bucket ke kebijakan bucket dan mengatur ulang ACL bucket Anda ke ACL pribadi default. Kami juga menyarankan Anda memigrasikan izin ACL objek ke kebijakan bucket dan mengedit kebijakan bucket yang memerlukan ACL selain ACL kontrol penuh pemilik bucket. Untuk informasi selengkapnya, lihat Prasyarat untuk menonaktifkan ACL.

Izin: Untuk menggunakan operasi ini, Anda harus memiliki `s3:PutBucketOwnershipControls` izin. Untuk informasi selengkapnya tentang izin Amazon S3, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang ingin Anda terapkan pengaturan Kepemilikan S3 Object.
3. Pilih tab Izin.
4. Di bawah Kepemilikan Objek, pilih Edit.
5. Di bawah Kepemilikan Objek, untuk menonaktifkan atau mengaktifkan ACL dan mengontrol kepemilikan objek yang diunggah di bucket Anda, pilih salah satu pengaturan berikut ini:

ACL dinonaktifkan

- Pemilik bucket diberlakukan–ACL dinonaktifkan, dan pemilik bucket secara otomatis memiliki dan memiliki kontrol penuh atas setiap objek di bucket. ACL tidak lagi memengaruhi izin terhadap data di bucket S3. Bucket tersebut menggunakan kebijakan untuk menentukan kontrol akses.

  Untuk mewajibkan semua bucket baru dibuat dengan ACL dinonaktifkan menggunakan IAM atau AWS Organizations kebijakan, lihat. Menonaktifkan ACL untuk semua bucket baru (pemilik bucket diberlakukan)

ACL diaktifkan

- Pemilik bucket yang dipilih—Pemilik bucket memiliki dan diberikan kendali penuh atas objek baru yang ditulis akun lain ke bucket dengan ACL `bucket-owner-full-control` yang dibatasi.

  Jika Anda menerapkan pengaturan pemilik bucket yang dipilih, agar semua unggahan Amazon S3 menyertakan ACL `bucket-owner-full-control` yang terekam, Anda dapat menambahkan kebijakan bucket yang hanya mengizinkan unggahan objek yang menggunakan ACL ini.
- Penulis objek — Akun AWS Yang mengunggah objek memiliki objek, memiliki kontrol penuh atasnya, dan dapat memberikan pengguna lain akses ke sana melalui ACL.

6. Pilih Simpan.

Menggunakan AWS CLI

Untuk menerapkan pengaturan Kepemilikan Objek untuk bucket yang ada, gunakan perintah `put-bucket-ownership-controls` dengan parameter `--ownership-controls`. Nilai yang valid untuk kepemilikan adalah`BucketOwnerEnforced`, `BucketOwnerPreferred`, atau `ObjectWriter`.

Contoh ini menerapkan pengaturan yang diberlakukan pemilik Bucket untuk bucket yang ada dengan menggunakan: AWS CLI

```
aws s3api put-bucket-ownership-controls --bucket DOC-EXAMPLE-BUCKET --ownership-
controls="Rules=[{ObjectOwnership=BucketOwnerEnforced}]"
```

Untuk informasi selengkapnya tentang `put-bucket-ownership-controls`, lihat put-bucket-ownership-controls di Panduan Pengguna AWS Command Line Interface .

Menggunakan AWS SDK for Java

Contoh ini menerapkan pengaturan `BucketOwnerEnforced` untuk Kepemilikan Objek pada bucket yang ada dengan menggunakan AWS SDK for Java:

```
// Build the ObjectOwnership for BucketOwnerEnforced
OwnershipControlsRule rule = OwnershipControlsRule.builder()
       .objectOwnership(ObjectOwnership.BucketOwnerEnforced)
       .build();

OwnershipControls ownershipControls = OwnershipControls.builder()
          .rules(rule)
          .build()

 // Build the PutBucketOwnershipControlsRequest
 PutBucketOwnershipControlsRequest putBucketOwnershipControlsRequest =
       PutBucketOwnershipControlsRequest.builder()
               .bucket(BUCKET_NAME)
               .ownershipControls(ownershipControls)
               .build();

 // Send the request to Amazon S3
 s3client.putBucketOwnershipControls(putBucketOwnershipControlsRequest);
```

Menggunakan AWS CloudFormation

AWS CloudFormation Untuk menggunakan pengaturan Kepemilikan Objek untuk bucket yang ada, lihat AWS::S3::Bucket OwnershipControlsdi Panduan AWS CloudFormation Pengguna.

Penggunaan API REST

Untuk menggunakan API REST untuk menerapkan pengaturan Kepemilikan Objek ke bucket S3 yang ada, gunakan `PutBucketOwnershipControls`. Untuk informasi selengkapnya, lihat PutBucketOwnershipControls dalam Referensi API Amazon Simple Storage Service.

Langkah selanjutnya: Setelah menerapkan pengaturan pilihan pemilik Bucket yang diberlakukan atau pemilik bucket untuk Kepemilikan Objek, Anda dapat mengambil langkah-langkah berikut:

- Pemilik bucket yang diberlakukan–Mengharuskan semua bucket baru dibuat dengan ACL dinonaktifkan menggunakan kebijakan IAM atau Organisasi.

- Pemilik bucket yang dipilih–Menambahkan kebijakan bucket S3 untuk ACL terekam `bucket-owner-full-control` untuk semua unggahan objek ke bucket Anda.

Melihat pengaturan Kepemilikan Objek untuk bucket S3

Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk menonaktifkan daftar kontrol akses (ACL) dan mengambil kepemilikan setiap objek di bucket Anda, menyederhanakan manajemen akses untuk data yang disimpan di Amazon S3. Secara default, Kepemilikan Objek S3 diatur ke pengaturan yang diberlakukan pemilik Bucket, dan ACL dinonaktifkan untuk bucket baru. Dengan menonaktifkan ACL, pemilik bucket memiliki setiap objek di bucket dan mengelola akses ke data secara eksklusif menggunakan kebijakan manajemen akses. Kami menyarankan agar ACL dinonaktifkan, kecuali dalam keadaan yang tidak biasa ketika Anda perlu mengontrol akses untuk setiap objek secara individual.

Kepemilikan Objek memiliki tiga pengaturan yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda, dan untuk menonaktifkan atau mengaktifkan ACL:

ACL dinonaktifkan

- Pemilik bucket yang diberlakukan (default)–ACL dinonaktifkan, dan pemilik bucket secara otomatis memilikinya, serta mendapatkan kontrol penuh atas setiap objek di dalam bucket. ACL tidak lagi memengaruhi izin terhadap data di bucket S3. Bucket tersebut menggunakan kebijakan untuk menentukan kontrol akses.

ACL diaktifkan

- Pemilik bucket yang dipilih—Pemilik bucket memiliki dan diberikan kendali penuh atas objek baru yang ditulis akun lain ke bucket dengan ACL `bucket-owner-full-control` yang dibatasi.
- Penulis objek — Akun AWS Yang mengunggah objek memiliki objek, memiliki kontrol penuh atasnya, dan dapat memberikan pengguna lain akses ke sana melalui ACL.

Anda dapat melihat pengaturan Kepemilikan S3 Object untuk bucket Amazon S3. Untuk menyetel Kepemilikan Objek untuk bucket baru, lihatMengatur Kepemilikan Objek saat membuat bucket. Untuk mengatur Kepemilikan Objek untuk bucket yang ada, lihat Menyetel Kepemilikan Objek pada bucket yang ada.

Izin: Untuk menggunakan operasi ini, Anda harus memiliki izin `s3:GetBucketOwnershipControls`. Untuk informasi selengkapnya tentang izin Amazon S3, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang ingin Anda terapkan pengaturan Kepemilikan Objek.
3. Pilih tab Izin.
4. Di bawah Kepemilikan Objek, Anda dapat melihat pengaturan Kepemilikan Objek untuk bucket Anda.

Menggunakan AWS CLI

Untuk mengambil pengaturan Kepemilikan Objek S3 untuk bucket S3, gunakan perintah. get-bucket-ownership-controls AWS CLI

```
aws s3api get-bucket-ownership-controls --bucket DOC-EXAMPLE-BUCKET
```

Penggunaan API REST

Untuk mengambil pengaturan Kepemilikan Objek untuk bucket S3, gunakan operasi API `GetBucketOwnershipControls`. Untuk informasi selengkapnya, lihat GetBucketOwnershipControls.

Menonaktifkan ACL untuk semua bucket baru dan menegakkan Kepemilikan Objek

Kami sarankan Anda menonaktifkan ACL di bucket Amazon S3. Anda dapat melakukannya dengan menerapkan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan S3 Object. Ketika Anda menerapkan pengaturan ini, ACL dinonaktifkan dan Anda secara otomatis memiliki dan memiliki kontrol penuh atas semua objek di bucket Anda. Untuk mewajibkan semua bucket baru dibuat dengan ACL dinonaktifkan, gunakan kebijakan AWS Identity and Access Management (IAM) atau kebijakan kontrol AWS Organizations layanan (SCP), seperti yang dijelaskan di bagian berikutnya.

Untuk menerapkan kepemilikan objek untuk objek baru tanpa menonaktifkan ACL, Anda dapat menerapkan pengaturan pilihan pemilik Bucket. Saat menerapkan pengaturan ini, kami sangat

menyarankan agar Anda memperbarui kebijakan bucket untuk `bucket-owner-full-control` mewajibkan ACL yang terekam untuk semua permintaan PUT ke bucket Anda. Pastikan Anda juga memperbarui klien Anda untuk mengirim ACL terekam `bucket-owner-full-control` ke bucket Anda dari akun lain.

Topik

- Menonaktifkan ACL untuk semua bucket baru (pemilik bucket diberlakukan)
- Memerlukan ACL bucket-owner-full-control kalengan untuk operasi Amazon PUT S3 (lebih disukai pemilik ember)

Menonaktifkan ACL untuk semua bucket baru (pemilik bucket diberlakukan)

Contoh berikut kebijakan IAM menolak izin `s3:CreateBucket` untuk pengguna atau peran IAM tertentu kecuali pengaturan yang diberlakukan pemilik Bucket diterapkan untuk Kepemilikan Objek. Pasangan kunci-nilai di dalam blok `Condition` menentukan `s3:x-amz-object-ownership` sebagai kunci, dan pengaturan `BucketOwnerEnforced` sebagai nilainya. Dengan kata lain, pengguna IAM hanya dapat membuat bucket jika mereka menyetel pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek dan menonaktifkan ACL. Anda juga dapat menggunakan kebijakan ini sebagai SCP batas untuk organisasi Anda. AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RequireBucketOwnerFullControl",
            "Action": "s3:CreateBucket",
            "Effect": "Deny",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-object-ownership": "BucketOwnerEnforced"
                }
            }
        }
    ]
}
```

Memerlukan ACL bucket-owner-full-control kalengan untuk operasi Amazon **PUT** S3 (lebih disukai pemilik ember)

Dengan pengaturan pilihan pemilik Bucket untuk Kepemilikan Objek, Anda, sebagai pemilik bucket, memiliki dan memiliki kendali penuh atas objek baru yang ditulis akun lain ke bucket Anda dengan ACL terekam `bucket-owner-full-control`. Namun, jika akun lain menulis objek ke bucket Anda tanpa ACL terekam`bucket-owner-full-control`, penulis objek mempertahankan akses kontrol penuh. Anda, sebagai pemilik bucket, dapat menerapkan kebijakan bucket yang mengizinkan penulisan hanya jika mereka menentukan ACL terekam `bucket-owner-full-control`.

> **Note**
>
> Jika ACL dinonaktifkan dengan pengaturan yang diberlakukan pemilik Bucket, Anda, sebagai pemilik bucket, secara otomatis memiliki dan memiliki kendali penuh atas semua objek di bucket Anda. Anda tidak perlu menggunakan bagian ini untuk memperbarui kebijakan bucket guna menegakkan kepemilikan objek bagi pemilik bucket.

Kebijakan bucket berikut ini menentukan akun tersebut *111122223333* dapat unggah objek ke *DOC-EXAMPLE-BUCKET* hanya ketika objek ACL diatur ke `bucket-owner-full-control`. Pastikan untuk mengganti *111122223333* dengan akun Anda, dan *DOC-EXAMPLE-BUCKET* dengan nama bucket Anda.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "Only allow writes to my bucket with bucket owner full control",
         "Effect": "Allow",
         "Principal": {
            "AWS": [
               "arn:aws:iam::111122223333:user/ExampleUser"
            ]
         },
         "Action": [
            "s3:PutObject"
         ],
         "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
         "Condition": {
            "StringEquals": {
```

```
                "s3:x-amz-acl": "bucket-owner-full-control"
            }
        }
    }
  ]
}
```

Berikut ini adalah contoh operasi penyalinan yang menyertakan `bucket-owner-full-control` ACL terekam dengan menggunakan AWS Command Line Interface (AWS CLI).

```
aws s3 cp file.txt s3://DOC-EXAMPLE-BUCKET --acl bucket-owner-full-control
```

Setelah kebijakan bucket diberlakukan, jika klien tidak menyertakan ACL terekam `bucket-owner-full-control`, maka operasi akan gagal, dan pengunggah menerima kesalahan berikut:

Terjadi kesalahan (AccessDenied) saat memanggil PutObject operasi: Akses Ditolak.

> **Note**
>
> Jika klien perlu mengakses objek setelah mengunggah, maka Anda harus memberikan izin tambahan untuk akun pengunggah. Untuk informasi tentang bagaimana memberikan akun akses ke sumber daya Anda, lihat Panduan yang menggunakan kebijakan untuk mengelola akses ke sumber daya Amazon S3.

Pemecahan Masalah

Saat Anda menerapkan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan S3 Object, daftar kontrol akses (ACL) dinonaktifkan dan Anda, sebagai pemilik bucket, secara otomatis memiliki semua objek di bucket Anda. ACL tidak lagi memengaruhi izin untuk objek di bucket Anda. Anda dapat menggunakan kebijakan untuk memberikan izin. Semua permintaan PUT S3 harus menentukan ACL terekam `bucket-owner-full-control` atau tidak menentukan ACL sama sekali, atau permintaan ini akan gagal. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Jika ACL yang tidak valid ditentukan atau izin ACL bucket memberikan akses di luar Akun AWS Anda, Anda mungkin melihat respons kesalahan berikut.

AccessControlListNotSupported

Setelah Anda menerapkan pengaturan diberlakukan pemilik Bucket untuk Kepemilikan Objek, ACL akan dinonaktifkan. Permintaan untuk mengatur ACL atau memperbarui ACL gagal dengan `400` kesalahan dan mengembalikan kode AccessControlListNotSupportedkesalahan. Permintaan untuk membaca ACL masih didukung. Permintaan untuk membaca ACL selalu menampilkan respons yang menunjukkan kontrol penuh untuk pemilik bucket. Dalam operasi PUT Anda, Anda harus menentukan ACL kontrol penuh pemilik bucket atau tidak menentukan ACL. Jika tidak, operasi PUT Anda gagal.

Contoh `put-object` AWS CLI perintah berikut termasuk ACL `public-read` kalengan.

```
aws s3api put-object --bucket DOC-EXAMPLE-BUCKET --key object-key-name --body doc-
example-body --acl public-read
```

Jika bucket menggunakan pengaturan yang diberlakukan pemilik bucket untuk menonaktifkan ACL, maka operasi ini gagal, dan pengunggah menerima pesan galat berikut:

Terjadi kesalahan (AccessControlListNotSupported) saat memanggil PutObject operasi: Bucket tidak mengizinkan ACL

InvalidBucketAclWithObjectOwnership

Jika Anda ingin menerapkan pengaturan yang diberlakukan pemilik Bucket untuk menonaktifkan ACL, ACL bucket Anda harus memberikan kontrol penuh hanya kepada pemilik bucket. Bucket ACL Anda tidak dapat memberikan akses ke grup eksternal Akun AWS atau grup lainnya. Misalnya, jika `CreateBucket` permintaan Anda menetapkan diberlakukan pemilik Bucket dan menentukan ACL bucket yang menyediakan akses ke eksternal Akun AWS, permintaan Anda gagal dengan `400` kesalahan dan mengembalikan kode kesalahan. InvalidBucketAclWithObjectOwnership Demikian pula, jika permintaan `PutBucketOwnershipControls` Anda menetapkan pemilik Bucket yang diberlakukan pada bucket yang memiliki bucket ACL yang memberikan izin kepada orang lain, permintaan tersebut gagal.

Example : Bucket ACL yang ada memberikan akses baca publik

Misalnya, jika ACL bucket yang ada memberikan akses baca publik, Anda tidak dapat menerapkan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek hingga Anda memigrasikan izin ACL ini ke kebijakan bucket dan mengatur ulang ACL bucket Anda ke ACL pribadi default. Untuk informasi selengkapnya, lihat Prasyarat untuk menonaktifkan ACL.

Contoh bucket ACL ini memberikan akses baca publik:

```
{
    "Owner": {
        "ID": "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID"
    },
    "Grants": [
        {
            "Grantee": {
                "ID":
 "852b113e7a2f25102679df27bb0ae12b3f85be6BucketOwnerCanonicalUserID",
                "Type": "CanonicalUser"
            },
            "Permission": "FULL_CONTROL"
        },
        {
            "Grantee": {
                "Type": "Group",
                "URI": "http://acs.amazonaws.com/groups/global/AllUsers"
            },
            "Permission": "READ"
        }
    ]
}
```

`put-bucket-ownership-controls` AWS CLI Perintah contoh berikut menerapkan setelan diberlakukan pemilik Bucket untuk Kepemilikan Objek:

```
aws s3api put-bucket-ownership-controls --bucket DOC-EXAMPLE-BUCKET --ownership-
controls Rules=[{ObjectOwnership=BucketOwnerEnforced}]
```

Karena bucket ACL memberikan akses baca publik, permintaan gagal dan mengembalikan kode kesalahan berikut:

Terjadi kesalahan (InvalidBucketAclWithObjectOwnership) saat memanggil PutBucketOwnershipControls operasi: Bucket tidak dapat mengatur ACL dengan ObjectOwnership pengaturan BucketOwnerEnforced

Berbagi sumber daya lintas asal (CORS)

Cross-origin resource sharing (CORS) menentukan cara aplikasi web klien yang dimuat di dalam satu domain untuk berinteraksi dengan sumber daya di domain yang berbeda. Dengan dukungan CORS,

Anda dapat membangun aplikasi web sisi klien yang kaya dengan Amazon S3, dan secara selektif mengizinkan akses lintas asal ke sumber daya Amazon S3 Anda.

Bagian ini memberikan ikhtisar tentang CORS. Subtopik menjelaskan bagaimana Anda dapat mengaktifkan CORS menggunakan konsol Amazon S3, atau secara terprogram menggunakan Amazon S3 REST API dan SDK. AWS

Berbagi sumber daya lintas asal: Skenario kasus penggunaan

Berikut adalah contoh skenario untuk menggunakan CORS.

Skenario 1

Misalkan Anda menyelenggarakan situs web di bucket Amazon S3 dengan nama `website` sebagaimana dijelaskan dalam Hosting situs web statis menggunakan Amazon S3. Pengguna Anda memuat titik akhir situs web:

```
http://website.s3-website.us-east-1.amazonaws.com
```

Sekarang Anda ingin menggunakan JavaScript pada halaman web yang disimpan di bucket ini untuk dapat membuat permintaan GET dan PUT yang diautentikasi terhadap bucket yang sama dengan menggunakan titik akhir API Amazon S3 untuk bucket,. `website.s3.us-east-1.amazonaws.com` Browser biasanya akan memblokir JavaScript agar tidak mengizinkan permintaan tersebut, tetapi dengan CORS Anda dapat mengonfigurasi bucket Anda untuk secara eksplisit mengaktifkan permintaan lintas asal. `website.s3-website.us-east-1.amazonaws.com`

Skenario 2

Misalkan, Anda ingin menghosting font web dari bucket S3. Sekali lagi, browser memerlukan pemeriksaan CORS (juga disebut pemeriksaan preflight) untuk memuat font web. Anda harus mengonfigurasi bucket yang menghosting font web untuk memungkinkan asal dari mana pun untuk membuat permintaan ini.

Bagaimana Amazon S3 mengevaluasi konfigurasi CORS pada bucket?

Saat Amazon S3 menerima permintaan preflight dari browser, itu akan mengevaluasi konfigurasi CORS untuk bucket dan menggunakan aturan `CORSRule` pertama yang sesuai dengan permintaan browser yang masuk untuk mengaktifkan permintaan lintas asal. Agar aturan dapat dicocokkan, ketentuan berikut harus dipenuhi:

- Header `Origin` permintaan harus cocok dengan elemen `AllowedOrigin` lainnya.
- Metode permintaan (misalnya, GET atau PUT) atau header `Access-Control-Request-Method` jika terjadi permintaan `OPTIONS` preflight harus merupakan salah satu elemen `AllowedMethod`.
- Setiap header yang tercantum dalam header `Access-Control-Request-Headers` permintaan pada permintaan preflight harus sesuai dengan elemen `AllowedHeader`.

> **Note**
>
> ACL dan kebijakan terus berlaku saat Anda mengaktifkan CORS di bucket.

Bagaimana Titik Akses Lambda Objek mendukung CORS

Ketika S3 Lambda Objek menerima permintaan dari browser atau permintaannya menyertakan header `Origin`, S3 Lambda Objek selalu menambahkan bidang header `“AllowedOrigins":"*"`.

Untuk informasi lebih lanjut menggunakan CORS, lihat topik berikut.

Topik

- Konfigurasi CORS
- Berbagi sumber daya lintas asal (CORS)

Konfigurasi CORS

Untuk mengonfigurasi bucket Anda agar permintaan lintas asal dapat dilakukan, Anda membuat konfigurasi CORS. Konfigurasi CORS adalah dokumen dengan aturan yang mengidentifikasi asal-usul yang akan Anda izinkan untuk mengakses bucket Anda, operasi (metode HTTP) yang akan mendukung setiap asal, dan informasi kustom operasi lainnya. Anda dapat menambahkan hingga 100 aturan ke konfigurasi. Anda dapat menambahkan konfigurasi CORS sebagai sub-sumber daya `cors` ke bucket.

Jika Anda mengonfigurasi CORS di konsol S3, Anda harus menggunakan JSON untuk membuat konfigurasi CORS. Konsol S3 baru hanya mendukung konfigurasi JSON CORS.

Untuk informasi lebih lanjut tentang konfigurasi CORS dan unsur-unsur di dalamnya, lihat topik di bawah ini. Untuk petunjuk tentang cara menambahkan konfigurasi CORS, lihat Berbagi sumber daya lintas asal (CORS).

> **Important**
>
> Pada konsol S3, konfigurasi CORS harus merupakan JSON.

Topik

- Contoh 1
- Contoh 2
- AllowedMethod elemen
- AllowedOrigin elemen
- AllowedHeader elemen
- ExposeHeader elemen
- MaxAgeSeconds elemen

Contoh 1

Alih-alih mengakses situs web dengan menggunakan titik akhir situs web Amazon S3, Anda dapat menggunakan domain Anda sendiri, seperti `example1.com` untuk melayani konten Anda. Untuk informasi tentang menggunakan domain Anda sendiri, lihat Tutorial: Mengonfigurasi situs web statis menggunakan domain kustom yang terdaftar di Route 53.

Contoh konfigurasi `cors` berikut memiliki tiga aturan, yang ditentukan sebagai elemen `CORSRule`:

- Aturan pertama mengizinkan permintaan PUT, POST, dan DELETE lintas asal dari asal `http://www.example1.com`. Aturan ini juga memungkinkan semua header dalam permintaan OPSI preflight melalui header `Access-Control-Request-Headers`. Menanggapi permintaan OPTIONS preflight, Amazon S3 mengembalikan header yang diminta.
- Aturan kedua mengizinkan permintaan lintas asal yang sama seperti aturan pertama, tetapi aturan tersebut berlaku untuk asal lain, `http://www.example2.com`.
- Aturan ketiga memungkinkan permintaan GET lintas asal dari semua asal. Karakter wildcard `*` mengacu pada semua asal.

JSON

```
[
```

```
    {
        "AllowedHeaders": [
            "*"
        ],
        "AllowedMethods": [
            "PUT",
            "POST",
            "DELETE"
        ],
        "AllowedOrigins": [
            "http://www.example1.com"
        ],
        "ExposeHeaders": []
    },
    {
        "AllowedHeaders": [
            "*"
        ],
        "AllowedMethods": [
            "PUT",
            "POST",
            "DELETE"
        ],
        "AllowedOrigins": [
            "http://www.example2.com"
        ],
        "ExposeHeaders": []
    },
    {
        "AllowedHeaders": [],
        "AllowedMethods": [
            "GET"
        ],
        "AllowedOrigins": [
            "*"
        ],
        "ExposeHeaders": []
    }
]
```

XML

```
<CORSConfiguration>
```

```
  <CORSRule>
    <AllowedOrigin>http://www.example1.com</AllowedOrigin>

    <AllowedMethod>PUT</AllowedMethod>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>

    <AllowedHeader>*</AllowedHeader>
  </CORSRule>
  <CORSRule>
    <AllowedOrigin>http://www.example2.com</AllowedOrigin>

    <AllowedMethod>PUT</AllowedMethod>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>

    <AllowedHeader>*</AllowedHeader>
  </CORSRule>
  <CORSRule>
    <AllowedOrigin>*</AllowedOrigin>
    <AllowedMethod>GET</AllowedMethod>
  </CORSRule>
</CORSConfiguration>
```

Contoh 2

Konfigurasi CORS juga memungkinkan parameter konfigurasi opsional, seperti ditunjukkan dalam konfigurasi CORS berikut. Dalam contoh ini, konfigurasi CORS memungkinkan permintaan PUT, POST, dan DELETE lintas asal dari asal `http://www.example.com`.

JSON

```
[
    {
        "AllowedHeaders": [
            "*"
        ],
        "AllowedMethods": [
            "PUT",
            "POST",
            "DELETE"
        ],
```

```
        "AllowedOrigins": [
            "http://www.example.com"
        ],
        "ExposeHeaders": [
            "x-amz-server-side-encryption",
            "x-amz-request-id",
            "x-amz-id-2"
        ],
        "MaxAgeSeconds": 3000
    }
]
```

XML

```
<CORSConfiguration>
 <CORSRule>
   <AllowedOrigin>http://www.example.com</AllowedOrigin>
   <AllowedMethod>PUT</AllowedMethod>
   <AllowedMethod>POST</AllowedMethod>
   <AllowedMethod>DELETE</AllowedMethod>
   <AllowedHeader>*</AllowedHeader>
  <MaxAgeSeconds>3000</MaxAgeSeconds>
  <ExposeHeader>x-amz-server-side-encryption</
ExposeHeader>
  <ExposeHeader>x-amz-request-id</
ExposeHeader>
  <ExposeHeader>x-amz-id-2</ExposeHeader>
 </CORSRule>
</CORSConfiguration>
```

Elemen `CORSRule` dalam konfigurasi sebelumnya mencakup elemen opsional berikut:

- `MaxAgeSeconds`—Menentukan jumlah waktu dalam detik (dalam contoh ini, 3000) saat browser menyimpan respons Amazon S3 terhadap permintaan OPTIONS preflight untuk sumber daya yang ditentukan. Dengan menyimpan respons, browser tidak perlu mengirim permintaan preflight ke Amazon S3 jika permintaan awal akan diulang.
- `ExposeHeader`—Mengidentifikasi header respons (dalam contoh ini,, `x-amz-server-side-encryption`x`-amz-request-id`, dan`x-amz-id-2`) yang dapat diakses pelanggan dari aplikasi mereka (misalnya, dari objek). JavaScript `XMLHttpRequest`

AllowedMethod elemen

Dalam konfigurasi CORS, Anda dapat menentukan nilai berikut untuk elemen `AllowedMethod` lainnya.

- GET
- PUT
- POST
- HAPUS
- HEAD

AllowedOrigin elemen

Di elemen `AllowedOrigin`, Anda menentukan asal yang ingin Anda izinkan permintaan lintas domainnya, misalnya, `http://www.example.com`. String asal hanya dapat berisi satu karakter wildcard `*`, seperti `http://*.example.com`. Anda secara opsional dapat menentukan `*` sebagai asal untuk mengaktifkan semua asal untuk mengirim permintaan lintas asal. Anda juga dapat menentukan `https` untuk hanya memungkinkan asal yang aman.

AllowedHeader elemen

Elemen `AllowedHeader` menentukan header yang diizinkan dalam permintaan preflight melalui header `Access-Control-Request-Headers`. Setiap nama header di header `Access-Control-Request-Headers` harus cocok dengan entri yang sesuai pada aturan. Amazon S3 hanya akan mengirimkan header yang diizinkan dalam respons yang diminta. Untuk daftar sampel header yang dapat digunakan dengan permintaan ke Amazon S3, kunjungi Header Permintaan Umum dalam panduan Referensi API Amazon Simple Storage Service.

Setiap AllowedHeader string dalam aturan dapat berisi paling banyak satu karakter wildcard. Misalnya, `<AllowedHeader>x-amz-*</AllowedHeader>` akan mengaktifkan semua header kustom Amazon.

ExposeHeader elemen

Setiap `ExposeHeader` elemen mengidentifikasi header dalam respons yang Anda ingin pelanggan dapat mengakses dari aplikasi mereka (misalnya, dari JavaScript `XMLHttpRequest` objek). Untuk daftar header respons Amazon S3 umum, kunjungi Header Respons Umum dalam panduan Referensi API Amazon Simple Storage Service.

MaxAgeSeconds elemen

Elemen `MaxAgeSeconds` menentukan waktu dalam hitungan detik bahwa browser Anda dapat menyimpan respons untuk permintaan preflight sebagaimana diidentifikasi oleh sumber daya, metode HTTP, dan asal.

Berbagi sumber daya lintas asal (CORS)

Cross-origin resource sharing (CORS) menentukan cara aplikasi web klien yang dimuat di dalam satu domain untuk berinteraksi dengan sumber daya di domain yang berbeda. Dengan dukungan CORS, Anda dapat membangun aplikasi web sisi klien yang kaya dengan Amazon S3, dan secara selektif mengizinkan akses lintas asal ke sumber daya Amazon S3 Anda.

Bagian ini menunjukkan cara mengaktifkan CORS menggunakan konsol Amazon S3, Amazon S3 REST API, dan SDK. AWS Untuk mengonfigurasi bucket Anda untuk mengizinkan permintaan lintas asal, Anda menambahkan konfigurasi CORS ke bucket. Konfigurasi CORS adalah dokumen yang menetapkan aturan yang mengidentifikasi asal-usul yang akan Anda izinkan untuk mengakses bucket Anda, metode operasi (metode HTTP) yang didukung untuk setiap asal, dan informasi kustom operasi lainnya. Pada konsol S3, konfigurasi CORS harus dokumen JSON.

Misalnya konfigurasi CORS di JSON dan XL, lihat Konfigurasi CORS.

Menggunakan konsol S3

Bagian ini menjelaskan cara menggunakan konsol Amazon S3 untuk menambahkan konfigurasi berbagi sumber daya lintas asal (CORS) ke bucket S3.

Saat Anda mengaktifkan CORS pada bucket, daftar kontrol akses (ACL) dan kebijakan izin akses lainnya terus berlaku.

> **Important**
>
> Pada konsol S3 baru, konfigurasi CORS harus JSON. Untuk contoh konfigurasi CORS di JSON dan XML, lihat Konfigurasi CORS.

Untuk menambahkan konfigurasi CORS ke bucket S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di daftar Bucket, pilih nama bucket yang ingin Anda buatkan kebijakan kelompok.
3. Pilih Izin.
4. Di bagian Berbagi sumber daya lintas asal (CORS), pilih Edit.
5. Di kotak teks Editor konfigurasi CORS, ketik atau salin dan tempel konfigurasi CORS baru, atau edit konfigurasi yang ada.

   Konfigurasi CORS adalah file JSON. Teks yang Anda ketikkan di editor harus valid JSON. Untuk informasi selengkapnya, lihat Konfigurasi CORS.
6. Pilih Simpan perubahan.

> **Note**
>
> Amazon S3 menampilkan Amazon Resource Name (ARN) untuk bucket di samping judul Editor konfigurasi CORS. Untuk informasi selengkapnya tentang ARN, lihat Amazon Resource Names (ARN) dan AWS Service Namespaces di. Referensi Umum Amazon Web Services

Menggunakan AWS SDK

Anda dapat menggunakan AWS SDK untuk mengelola berbagi sumber daya lintas asal (CORS) untuk bucket. Untuk informasi selengkapnya tentang CORS, lihat Berbagi sumber daya lintas asal (CORS).

Contoh berikut:

- Membuat konfigurasi CORS dan mengatur konfigurasi pada bucket
- Membuka kembali konfigurasi dan memodifikasinya dengan menambahkan aturan
- Menambahkan konfigurasi yang telah dimodifikasi ke bucket
- Menghapus konfigurasi

Java

Example

Example

Untuk petunjuk tentang cara membuat dan menguji sampel kerja, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketCrossOriginConfiguration;
import com.amazonaws.services.s3.model.CORSRule;

import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class CORS {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        // Create two CORS rules.
        List<CORSRule.AllowedMethods> rule1AM = new
 ArrayList<CORSRule.AllowedMethods>();
        rule1AM.add(CORSRule.AllowedMethods.PUT);
        rule1AM.add(CORSRule.AllowedMethods.POST);
        rule1AM.add(CORSRule.AllowedMethods.DELETE);
        CORSRule rule1 = new
 CORSRule().withId("CORSRule1").withAllowedMethods(rule1AM)
                .withAllowedOrigins(Arrays.asList("http://*.example.com"));

        List<CORSRule.AllowedMethods> rule2AM = new
 ArrayList<CORSRule.AllowedMethods>();
        rule2AM.add(CORSRule.AllowedMethods.GET);
```

```
        CORSRule rule2 = new
CORSRule().withId("CORSRule2").withAllowedMethods(rule2AM)
                .withAllowedOrigins(Arrays.asList("*")).withMaxAgeSeconds(3000)
                .withExposedHeaders(Arrays.asList("x-amz-server-side-encryption"));

        List<CORSRule> rules = new ArrayList<CORSRule>();
        rules.add(rule1);
        rules.add(rule2);

        // Add the rules to a new CORS configuration.
        BucketCrossOriginConfiguration configuration = new
BucketCrossOriginConfiguration();
        configuration.setRules(rules);

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Add the configuration to the bucket.
            s3Client.setBucketCrossOriginConfiguration(bucketName, configuration);

            // Retrieve and display the configuration.
            configuration = s3Client.getBucketCrossOriginConfiguration(bucketName);
            printCORSConfiguration(configuration);

            // Add another new rule.
            List<CORSRule.AllowedMethods> rule3AM = new
ArrayList<CORSRule.AllowedMethods>();
            rule3AM.add(CORSRule.AllowedMethods.HEAD);
            CORSRule rule3 = new
CORSRule().withId("CORSRule3").withAllowedMethods(rule3AM)
                    .withAllowedOrigins(Arrays.asList("http://www.example.com"));

            rules = configuration.getRules();
            rules.add(rule3);
            configuration.setRules(rules);
            s3Client.setBucketCrossOriginConfiguration(bucketName, configuration);

            // Verify that the new rule was added by checking the number of rules in
the
            // configuration.
            configuration = s3Client.getBucketCrossOriginConfiguration(bucketName);
```

```
            System.out.println("Expected # of rules = 3, found " +
 configuration.getRules().size());

            // Delete the configuration.
            s3Client.deleteBucketCrossOriginConfiguration(bucketName);
            System.out.println("Removed CORS configuration.");

            // Retrieve and display the configuration to verify that it was
            // successfully deleted.
            configuration = s3Client.getBucketCrossOriginConfiguration(bucketName);
            printCORSConfiguration(configuration);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    private static void printCORSConfiguration(BucketCrossOriginConfiguration
 configuration) {
        if (configuration == null) {
            System.out.println("Configuration is null.");
        } else {
            System.out.println("Configuration has " +
 configuration.getRules().size() + " rules\n");

            for (CORSRule rule : configuration.getRules()) {
                System.out.println("Rule ID: " + rule.getId());
                System.out.println("MaxAgeSeconds: " + rule.getMaxAgeSeconds());
                System.out.println("AllowedMethod: " + rule.getAllowedMethods());
                System.out.println("AllowedOrigins: " + rule.getAllowedOrigins());
                System.out.println("AllowedHeaders: " + rule.getAllowedHeaders());
                System.out.println("ExposeHeader: " + rule.getExposedHeaders());
                System.out.println();
            }
        }
    }
}
```

.NET

Example

Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CORSTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            CORSConfigTestAsync().Wait();
        }
        private static async Task CORSConfigTestAsync()
        {
            try
            {
                // Create a new configuration request and add two rules
                CORSConfiguration configuration = new CORSConfiguration
                {
                    Rules = new System.Collections.Generic.List<CORSRule>
                        {
                          new CORSRule
                          {
                            Id = "CORSRule1",
                            AllowedMethods = new List<string> {"PUT", "POST",
 "DELETE"},
```

```
                            AllowedOrigins = new List<string> {"http://
*.example.com"}
                          },
                          new CORSRule
                          {
                            Id = "CORSRule2",
                            AllowedMethods = new List<string> {"GET"},
                            AllowedOrigins = new List<string> {"*"},
                            MaxAgeSeconds = 3000,
                            ExposeHeaders = new List<string> {"x-amz-server-side-
encryption"}
                          }
                        }
                };

                // Add the configuration to the bucket.
                await PutCORSConfigurationAsync(configuration);

                // Retrieve an existing configuration.
                configuration = await RetrieveCORSConfigurationAsync();

                // Add a new rule.
                configuration.Rules.Add(new CORSRule
                {
                    Id = "CORSRule3",
                    AllowedMethods = new List<string> { "HEAD" },
                    AllowedOrigins = new List<string> { "http://www.example.com" }
                });

                // Add the configuration to the bucket.
                await PutCORSConfigurationAsync(configuration);

                // Verify that there are now three rules.
                configuration = await RetrieveCORSConfigurationAsync();
                Console.WriteLine();
                Console.WriteLine("Expected # of rulest=3; found:{0}",
 configuration.Rules.Count);
                Console.WriteLine();
                Console.WriteLine("Pause before configuration delete. To continue,
 click Enter...");
                Console.ReadKey();

                // Delete the configuration.
                await DeleteCORSConfigurationAsync();
```

```
                // Retrieve a nonexistent configuration.
                configuration = await RetrieveCORSConfigurationAsync();
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
        }

        static async Task PutCORSConfigurationAsync(CORSConfiguration configuration)
        {

            PutCORSConfigurationRequest request = new PutCORSConfigurationRequest
            {
                BucketName = bucketName,
                Configuration = configuration
            };

            var response = await s3Client.PutCORSConfigurationAsync(request);
        }

        static async Task<CORSConfiguration> RetrieveCORSConfigurationAsync()
        {
            GetCORSConfigurationRequest request = new GetCORSConfigurationRequest
            {
                BucketName = bucketName

            };
            var response = await s3Client.GetCORSConfigurationAsync(request);
            var configuration = response.Configuration;
            PrintCORSRules(configuration);
            return configuration;
        }

        static async Task DeleteCORSConfigurationAsync()
        {
```

```
            DeleteCORSConfigurationRequest request = new
 DeleteCORSConfigurationRequest
            {
                BucketName = bucketName
            };
            await s3Client.DeleteCORSConfigurationAsync(request);
        }

        static void PrintCORSRules(CORSConfiguration configuration)
        {
            Console.WriteLine();

            if (configuration == null)
            {
                Console.WriteLine("\nConfiguration is null");
                return;
            }

            Console.WriteLine("Configuration has {0} rules:",
 configuration.Rules.Count);
            foreach (CORSRule rule in configuration.Rules)
            {
                Console.WriteLine("Rule ID: {0}", rule.Id);
                Console.WriteLine("MaxAgeSeconds: {0}", rule.MaxAgeSeconds);
                Console.WriteLine("AllowedMethod: {0}", string.Join(", ",
 rule.AllowedMethods.ToArray()));
                Console.WriteLine("AllowedOrigins: {0}", string.Join(", ",
 rule.AllowedOrigins.ToArray()));
                Console.WriteLine("AllowedHeaders: {0}", string.Join(", ",
 rule.AllowedHeaders.ToArray()));
                Console.WriteLine("ExposeHeader: {0}", string.Join(", ",
 rule.ExposeHeaders.ToArray()));
            }
        }
    }
}
```

Penggunaan API REST

Untuk mengatur konfigurasi CORS pada bucket Anda, Anda dapat menggunakan AWS Management Console. Jika aplikasi Anda memerlukannya, Anda juga dapat mengirim permintaan REST secara

langsung. Bagian berikut dalam Referensi API Amazon Simple Storage Service menguraikan tindakan API REST terkait konfigurasi CORS:

- PutBucketCors
- GetBucketCors
- DeleteBucketCors
- OPTIONS object

Pencatatan log dan pemantauan di Amazon S3

Pemantauan adalah bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja Amazon S3 dan solusi Anda AWS . Anda harus mengumpulkan data pemantauan dari semua bagian AWS solusi Anda sehingga Anda dapat lebih mudah men-debug kegagalan multi-titik jika terjadi. AWS menyediakan beberapa alat untuk memantau sumber daya Amazon S3 Anda dan menanggapi potensi insiden.

Untuk informasi selengkapnya, lihat Pemantauan Amazon S3.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

CloudWatch Alarm Amazon

Menggunakan CloudWatch alarm Amazon, Anda menonton satu metrik selama periode waktu yang Anda tentukan. Jika metrik melebihi ambang batas tertentu, pemberitahuan akan dikirim ke topik atau AWS Auto Scaling kebijakan Amazon SNS. CloudWatch alarm tidak memanggil tindakan karena mereka berada dalam keadaan tertentu. Sebaliknya, kondisi tersebut harus diubah dan dipertahankan selama periode tertentu. Untuk informasi selengkapnya, lihat Memantau metrik dengan Amazon CloudWatch.

AWS CloudTrail Log

CloudTrail menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di Amazon S3. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat ke Amazon S3, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan. Untuk informasi selengkapnya, lihat Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail.

Amazon GuardDuty

Amazon GuardDuty adalah layanan deteksi ancaman yang terus memantau akun, container, beban kerja, dan data di AWS lingkungan Anda untuk mengidentifikasi potensi ancaman atau risiko keamanan terhadap bucket S3 Anda. GuardDuty juga menyediakan konteks yang kaya tentang ancaman yang dideteksinya. GuardDuty memantau log AWS CloudTrail manajemen

untuk ancaman dan memunculkan informasi keamanan yang relevan. Misalnya, GuardDuty akan mencakup faktor permintaan API, seperti pengguna yang membuat permintaan, lokasi permintaan dibuat, dan API spesifik yang diminta, yang mungkin tidak biasa di lingkungan Anda. GuardDuty S3 Protection memantau peristiwa data S3 yang dikumpulkan oleh CloudTrail dan mengidentifikasi perilaku yang berpotensi anomali dan berbahaya di semua bucket S3 di lingkungan Anda.

Log Akses Amazon S3

Log akses server memberikan catatan terperinci tentang permintaan yang dibuat ke bucket. Log akses server bermanfaat untuk berbagai macam aplikasi. Misalnya, informasi log akses dapat berguna dalam audit keamanan dan akses. Untuk informasi selengkapnya, lihat Pencatatan permintaan dengan pencatatan akses server.

AWS Trusted Advisor

Trusted Advisor mengacu pada praktik terbaik yang dipelajari dari melayani ratusan ribu AWS pelanggan. Trusted Advisor memeriksa AWS lingkungan Anda dan kemudian membuat rekomendasi ketika ada peluang untuk menghemat uang, meningkatkan ketersediaan dan kinerja sistem, atau membantu menutup kesenjangan keamanan. Semua AWS pelanggan memiliki akses ke lima Trusted Advisor cek. Pelanggan dengan paket dukungan Bisnis atau Perusahaan dapat melihat semua Trusted Advisor pemeriksaan.

Trusted Advisor memiliki pemeriksaan terkait Amazon S3 berikut:

- Konfigurasi pencatatan log bucket Amazon S3.
- Pemeriksaan keamanan untuk bucket Amazon S3 yang memiliki izin akses terbuka.
- Pengecekan toleransi kesalahan untuk bucket Amazon S3 yang tidak memiliki Penentuan Versi yang diaktifkan, atau memiliki Penentuan Versi yang ditangguhkan.

Untuk informasi selengkapnya, lihat AWS Trusted Advisor di AWS Support Panduan Pengguna.

Praktik keamanan terbaik berikut juga membahas pencatatan dan pemantauan:

- Identify and audit all your Amazon S3 buckets
- Implement monitoring using Amazon Web Services monitoring tools
- Aktifkan AWS Config
- Enable Amazon S3 server access logging
- Use CloudTrail

- Monitor Amazon Web Services security advisories

Validasi Kepatuhan untuk Amazon S3

Keamanan dan kepatuhan Amazon S3 dinilai oleh auditor pihak ketiga sebagai bagian dari beberapa program AWS kepatuhan, termasuk yang berikut:

- Kontrol Sistem dan Organisasi (SOC)
- Standar Keamanan Data Industri Kartu Pembayaran (PCI DSS)
- Program Manajemen Risiko dan Otorisasi Federal (FedRAMP)
- Undang-Undang Akuntabilitas dan Portabilitas Asuransi Kesehatan (HIPAA)

AWS menyediakan daftar AWS layanan yang sering diperbarui dalam lingkup program kepatuhan khusus di AWS Layanan dalam Lingkup oleh Program Kepatuhan.

Laporan audit pihak ketiga tersedia untuk Anda unduh AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifak.

Untuk informasi selengkapnya tentang program AWS kepatuhan, lihat Program AWS Kepatuhan.

Tanggung jawab kepatuhan Anda saat menggunakan Amazon S3 ditentukan oleh sensitivitas data Anda, tujuan kepatuhan organisasi Anda, serta undang-undang dan peraturan yang berlaku. Jika penggunaan Amazon S3 Anda tunduk pada kepatuhan standar seperti HIPAA, PCI, atau FedRAMP, AWS menyediakan sumber daya untuk membantu:

- Panduan Memulai Cepat Keamanan dan Kepatuhan yang membahas pertimbangan arsitektur dan langkah-langkah untuk menerapkan lingkungan dasar yang berfokus pada keamanan dan kepatuhan. AWS
- Arsitektur untuk Keamanan dan Kepatuhan HIPAA menguraikan bagaimana perusahaan menggunakan AWS untuk membantu mereka memenuhi persyaratan HIPAA.
- AWS Sumber Daya Kepatuhan menyediakan beberapa buku kerja dan panduan berbeda yang mungkin berlaku untuk industri dan lokasi Anda.
- AWS Config dapat digunakan untuk menilai seberapa baik konfigurasi sumber daya Anda telah mematuhi praktik internal, pedoman industri, dan peraturan yang berlaku.
- AWS Security Hub memberi Anda pandangan komprehensif tentang status keamanan Anda di dalamnya AWS dan membantu Anda memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik.

- Menggunakan Kunci Objek S3 dapat membantu Anda memenuhi persyaratan teknis dari regulator layanan keuangan (seperti SEC, FINRA, dan CFTC) yang memerlukan penyimpanan data “tulis sekali, baca banyak” (WORM) untuk jenis informasi pembukuan dan pencatatan tertentu.
- Inventaris Amazon S3 dapat membantu Anda mengaudit dan melaporkan replikasi dan status enkripsi objek Anda untuk kebutuhan bisnis, kepatuhan, dan regulasi.

Ketahanan dalam Amazon S3

Infrastruktur AWS global dibangun di sekitar Wilayah dan Zona Ketersediaan. Wilayah AWS menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan. Zona Ketersediaan ini menawarkan cara efektif untuk merancang dan mengoperasikan aplikasi dan basis data. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur pusat data tunggal atau multi tradisional. Jika Anda secara khusus perlu mereplikasi data Anda pada jarak geografis yang lebih besar, Anda dapat menggunakanMereplikasi ikhtisar objek, yang memungkinkan penyalinan objek secara otomatis dan asinkron di seluruh ember dalam berbagai jenis. Wilayah AWS

Masing-masing Wilayah AWS memiliki beberapa Availability Zone. Anda dapat menerapkan aplikasi Anda di beberapa Zona Ketersediaan di Wilayah yang sama untuk toleransi kesalahan dan latensi rendah. Zona Ketersediaan saling terhubung satu sama lain dengan jaringan serat optik pribadi yang cepat, memungkinkan Anda untuk merancang aplikasi secara mudah yang menerapkan pengalihan otomatis antara Zona Ketersediaan tanpa gangguan.

Untuk informasi selengkapnya tentang Wilayah AWS dan Availability Zone, lihat Infrastruktur AWS Global.

Selain infrastruktur AWS global, Amazon S3 menawarkan beberapa fitur untuk membantu mendukung ketahanan data dan kebutuhan cadangan Anda.

Konfigurasi siklus hidup

Konfigurasi siklus aktif adalah serangkaian aturan yang menentukan tindakan yang diterapkan Amazon S3 pada sekelompok objek. Dengan aturan konfigurasi siklus aktif, Anda dapat memberi tahu Amazon S3 untuk melakukan transisi objek ke kelas penyimpanan yang lebih murah, mengarsipkan, atau menghapusnya. Untuk informasi selengkapnya, lihat Mengelola siklus hidup penyimpanan Anda.

Penentuan Versi

Penentuan Versi adalah cara menyimpan beberapa varian objek dalam bucket yang sama. Anda dapat menggunakan Penentuan Versi untuk menyimpan, mengambil, dan memulihkan setiap versi dari setiap objek yang disimpan dalam bucket Amazon S3. Dengan Penentuan Versi, Anda dapat dengan mudah memulihkan dari tindakan pengguna yang tidak diinginkan, serta kegagalan aplikasi. Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.

Kunci Objek S3

Anda dapat menggunakan Kunci Objek S3 untuk menyimpan objek menggunakan model tulis-sekali-baca-banyak (WORM). Dengan Kunci Objek S3, Anda dapat mencegah objek agar tidak dihapus atau ditimpa selama jangka waktu tertentu atau tanpa batas waktu. Kunci Objek S3 memungkinkan Anda memenuhi persyaratan peraturan yang memerlukan penyimpanan WORM atau hanya untuk menambahkan lapisan perlindungan tambahan terhadap perubahan dan penghapusan objek. Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek S3.

Kelas penyimpanan

Amazon S3 menawarkan serangkaian kelas penyimpanan untuk dipilih tergantung pada persyaratan beban kerja Anda. Kelas penyimpanan S3 Standard-IA dan S3 One Zone-IA dirancang untuk data yang Anda akses sekitar sebulan sekali dan membutuhkan akses milidetik. Kelas penyimpanan S3 Glacier Instant Retrieval dirancang untuk data arsip berumur panjang yang diakses dengan akses milidetik yang Anda akses sekitar seperempat sekali. Untuk data arsip yang tidak memerlukan akses langsung, seperti backup, Anda dapat menggunakan kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive. Untuk informasi selengkapnya, lihat Menggunakan kelas penyimpanan Amazon S3.

Praktik keamanan terbaik berikut juga menangani ketahanan:

- Enable versioning
- Consider Amazon S3 cross-region replication
- Identify and audit all your Amazon S3 buckets

Enkripsi cadangan Amazon S3

Jika Anda menyimpan cadangan menggunakan Amazon S3, enkripsi cadangan Anda tergantung pada konfigurasi bucket tersebut. Amazon S3 menyediakan cara untuk menetapkan perilaku enkripsi default untuk bucket S3. Anda dapat menetapkan enkripsi default pada bucket sehingga semua objek dienkripsi saat disimpan dalam bucket. Enkripsi default mendukung kunci yang disimpan di AWS KMS (SSE-KMS). Untuk informasi selengkapnya, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3.

Untuk informasi lebih lanjut tentang Penentuan Versi dan Kunci Objek, lihat topik berikut: Menggunakan Penentuan Versi dalam bucket S3 Menggunakan Kunci Objek S3

Keamanan infrastruktur di Amazon S3

Sebagai layanan terkelola, Amazon S3 dilindungi oleh prosedur keamanan jaringan AWS global yang dijelaskan dalam pilar keamanan Kerangka Well-Architected AWS .

Akses ke Amazon S3 melalui jaringan adalah melalui API yang AWS dipublikasikan. Klien harus mendukung Keamanan Lapisan Pengangkutan (TLS) 1.2. Kami merekomendasikan juga mendukung TLS 1.3. (Untuk informasi selengkapnya tentang rekomendasi ini, lihat Koneksi AWS cloud yang lebih cepat dengan TLS 1.3 di Blog AWS Keamanan.) Klien juga harus mendukung suite cipher dengan perfect forward secrecy (PFS) seperti Ephemeral Diffie-Hellman (DHE) atau Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Selain itu, permintaan harus ditandatangani menggunakan AWS Signature V4 atau AWS Signature V2, yang memerlukan kredensyal yang valid untuk diberikan.

API ini dapat dihubungi dari lokasi jaringan mana pun. Namun, Amazon S3 mendukung kebijakan akses berbasis sumber daya, yang dapat mencakup pembatasan berdasarkan alamat IP sumber. Anda juga dapat menggunakan kebijakan bucket Amazon S3 untuk mengendalikan akses dari titik akhir cloud privat virtual (VPC) tertentu atau VPC tertentu. Secara efektif, ini mengisolasi akses jaringan ke bucket Amazon S3 tertentu hanya dari VPC tertentu dalam jaringan. AWS Untuk informasi selengkapnya, lihat Megendalikan akses dari titik akhir VPC dengan kebijakan bucket.

Praktik terbaik keamanan berikut juga membahas keamanan infrastruktur di Amazon S3:

- Consider VPC endpoints for Amazon S3 access
- Identify and audit all your Amazon S3 buckets

Analisis konfigurasi dan kerentanan di Amazon S3

AWS menangani tugas-tugas keamanan dasar seperti sistem operasi tamu (OS) dan patching database, konfigurasi firewall, dan pemulihan bencana. Prosedur ini telah ditinjau dan disertifikasi oleh pihak ketiga yang sesuai. Untuk detail selengkapnya, lihat sumber daya berikut:

- Validasi Kepatuhan untuk Amazon S3
- Model Tanggung Jawab Bersama
- Amazon Web Services: Gambaran Umum Proses Keamanan

Praktik terbaik keamanan berikut juga membahas konfigurasi dan analisis kerentanan di Amazon S3:

- Identify and audit all your Amazon S3 buckets
- Aktifkan AWS Config

Praktik Terbaik Keamanan untuk Amazon S3

Amazon S3 menyediakan sejumlah fitur keamanan untuk dipertimbangkan ketika Anda mengembangkan dan menerapkan kebijakan keamanan Anda sendiri. Praktik terbaik berikut adalah pedoman umum dan tidak mewakili solusi keamanan yang lengkap. Karena praktik terbaik ini mungkin tidak sesuai atau tidak memadai untuk lingkungan Anda, perlakukan itu sebagai rekomendasi yang bermanfaat, bukan sebagai resep.

Topik

- Praktik terbaik keamanan Amazon S3
- Praktik Terbaik Pemantauan dan Audit Amazon S3

Praktik terbaik keamanan Amazon S3

Praktik terbaik berikut untuk Amazon S3 dapat membantu mencegah insiden keamanan.

Menonaktifkan daftar kontrol akses (ACL)

Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda dan untuk menonaktifkan atau mengaktifkan ACL. Secara default, Kepemilikan Objek disetel ke pengaturan diberlakukan pemilik Bucket dan semua ACL dinonaktifkan. Ketika ACL dinonaktifkan, pemilik bucket memiliki semua objek di bucket dan mengelola akses ke data secara eksklusif menggunakan kebijakan manajemen akses.

Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan daftar kontrol akses (ACL). Sebaiknya Anda menonaktifkan ACL, kecuali dalam keadaan yang tidak biasa yang mengharuskan Anda mengontrol akses untuk setiap objek secara individual. Untuk menonaktifkan ACL dan mengambil kepemilikan setiap objek di bucket Anda, terapkan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan S3 Object. Saat menonaktifkan ACL, Anda dapat dengan mudah memelihara bucket dengan objek yang diunggah oleh berbagai objek Akun AWS.

Ketika ACL dinonaktifkan, kontrol akses untuk data Anda didasarkan pada kebijakan, seperti berikut ini:

- AWS Identity and Access Management (IAM) kebijakan pengguna
- Kebijakan bucket S3

- Kebijakan titik akhir cloud privat virtual (VPC)
- AWS Organizations kebijakan kontrol layanan (SCP)

Menonaktifkan ACL menyederhanakan manajemen izin dan audit. ACL dinonaktifkan untuk bucket baru secara default. Anda juga dapat menonaktifkan ACL untuk bucket yang ada. Jika Anda memiliki bucket yang sudah memiliki objek di dalamnya, setelah Anda menonaktifkan ACL, ACL objek dan bucket tidak lagi menjadi bagian dari proses evaluasi akses. Sebaliknya, akses diberikan atau ditolak berdasarkan kebijakan.

Sebelum menonaktifkan ACLs, pastikan Anda melakukan hal berikut ini:

- Tinjau kebijakan bucket Anda untuk memastikan bahwa kebijakan tersebut mencakup semua cara yang ingin Anda berikan akses ke bucket di luar akun Anda.
- Setel ulang ACL bucket Anda ke default (kontrol penuh ke pemilik bucket).

Setelah Anda menonaktifkan ACL, perilaku berikut terjadi:

- Bucket Anda hanya menerima permintaan `PUT` yang tidak menentukan ACL atau permintaan `PUT` dengan ACL kontrol penuh pemilik bucket. ACL ini termasuk ACL terekam `bucket-owner-full-control` atau bentuk setara dari ACL ini yang diekspresikan dalam XML.
- Aplikasi yang ada yang mendukung ACL kontrol penuh pemilik bucket tidak akan berdampak.
- `PUT`permintaan yang berisi ACL lain (misalnya, hibah khusus untuk tertentu Akun AWS) gagal dan mengembalikan kode status HTTP `400 (Bad Request)` dengan kode kesalahan. `AccessControlListNotSupported`

Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Pastikan bucket Amazon S3 Anda menggunakan kebijakan yang benar dan tidak dapat diakses publik

Kecuali jika Anda secara tegas meminta siapa pun di internet untuk dapat membaca atau menulis ke bucket S3 Anda, pastikan bucket S3 Anda tidak tersedia untuk umum. Berikut ini adalah beberapa langkah yang dapat Anda lakukan untuk memblokir akses publik:

- Gunakan Blokir Akses Publik S3. Dengan Blokir Akses Publik S3, Anda dapat dengan mudah menyiapkan kontrol terpusat untuk membatasi akses publik ke sumber daya Amazon S3 Anda. Kontrol terpusat ini ditegakkan terlepas dari bagaimana sumber daya dibuat. Untuk informasi selengkapnya, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

- Identifikasi kebijakan bucket Amazon S3 yang mengizinkan identitas wildcard seperti `"Principal": "*"` (yang secara efektif berarti “siapa pun”). Cari juga kebijakan yang memungkinkan tindakan wildcard `"*"` (yang secara efektif memungkinkan pengguna melakukan tindakan apa pun di bucket Amazon S3).
- Demikian pula, cari daftar kontrol akses bucket Amazon S3 (ACL) yang menyediakan akses baca, tulis, atau penuh ke “Semua Orang” atau “Setiap pengguna yang diautentikasi.” AWS
- Menggunakan operasi API `ListBuckets` untuk memindai semua bucket Amazon S3. Kemudian, gunakan `GetBucketAcl`, `GetBucketWebsite`, dan `GetBucketPolicy` untuk menentukan apakah setiap bucket memiliki kontrol akses yang sesuai dan konfigurasi yang sesuai.
- Gunakan AWS Trusted Advisor untuk memeriksa implementasi Amazon S3.
- Pertimbangkan untuk menerapkan kontrol detektif yang sedang berlangsung dengan menggunakan s3-bucket-public-read-prohibited dan s3-bucket-public-write-prohibited Aturan AWS Config yang dikelola.

Untuk informasi selengkapnya, lihat Identity and Access Management untuk Amazon S3.

Identifikasi potensi ancaman terhadap bucket Amazon S3 Anda dengan menggunakan Amazon GuardDuty

Amazon GuardDuty adalah layanan deteksi ancaman yang mengidentifikasi potensi ancaman terhadap akun, wadah, beban kerja, dan data di lingkungan Anda AWS . Dengan menggunakan model machine learning (ML), serta kemampuan deteksi anomali dan ancaman, Amazon GuardDuty terus memantau berbagai sumber data untuk mengidentifikasi dan memprioritaskan potensi risiko keamanan dan aktivitas berbahaya di lingkungan Anda. Saat Anda mengaktifkan GuardDuty, ia menawarkan deteksi ancaman untuk sumber data dasar yang mencakup peristiwa AWS CloudTrail manajemen, log aliran VPC, dan log DNS. Untuk memperluas deteksi ancaman ke peristiwa pesawat data di bucket S3, Anda dapat mengaktifkan fitur Perlindungan GuardDuty S3. Fitur ini mendeteksi ancaman seperti eksfiltrasi data dan akses mencurigakan ke bucket S3 melalui node Tor. GuardDutyjuga menetapkan pola dasar normal di lingkungan Anda dan ketika mengidentifikasi perilaku yang berpotensi anomali, ia memberikan informasi kontekstual untuk membantu Anda memulihkan ember atau kredensyal S3 yang berpotensi dikompromikan. AWS Untuk informasi lebih lanjut, lihat GuardDuty.

Terapkan akses hak akses paling rendah

Saat memberikan izin, Anda memutuskan siapa yang mendapatkan izin yang menjadi sumber daya Amazon S3. Anda mengaktifkan tindakan tertentu yang ingin Anda izinkan pada sumber daya tersebut. Oleh karena itu, kami menyarankan Anda hanya memberikan izin yang diperlukan

untuk melaksanakan tugas. Menerapkan akses hak akses paling rendah adalah hal mendasar dalam mengurangi risiko keamanan dan dampak yang dapat diakibatkan oleh kesalahan atau niat jahat.

Alat bantu berikut tersedia untuk menerapkan akses hak akses paling rendah:

- Tindakan kebijakan untuk Amazon S3 dan Batasan Izin untuk Entitas IAM
- Bagaimana Amazon S3 bekerja dengan IAM
- Gambaran umum daftar kontrol akses (ACL)
- Kebijakan Kontrol Layanan

Untuk panduan tentang apa yang harus dipertimbangkan ketika memilih satu atau beberapa mekanisme sebelumnya, lihat Identity and Access Management untuk Amazon S3.

Gunakan peran IAM untuk aplikasi dan Layanan AWS yang memerlukan akses Amazon S3

Agar aplikasi yang berjalan di Amazon EC2 atau lainnya Layanan AWS dapat mengakses sumber daya Amazon S3, mereka harus menyertakan kredensyal yang AWS valid dalam permintaan API mereka. AWS Sebaiknya jangan menyimpan AWS kredensialnya secara langsung di aplikasi atau instans Amazon EC2. Ini adalah kredensial jangka panjang yang tidak dirotasi secara otomatis dan dapat menimbulkan dampak bisnis yang signifikan jika dibobol.

Sebaliknya, gunakan peran IAM untuk mengelola kredensial sementara untuk aplikasi atau layanan yang perlu mengakses Amazon S3. Saat Anda menggunakan peran, Anda tidak perlu mendistribusikan kredensyal jangka panjang (seperti nama pengguna dan kata sandi atau kunci akses) ke instans Amazon EC2 Layanan AWS atau, seperti. AWS Lambda Peran ini menyediakan izin sementara yang dapat digunakan aplikasi saat mereka melakukan panggilan ke AWS sumber daya lain.

Untuk informasi selengkapnya, lihat topik berikut di Panduan Pengguna IAM:

- Peran IAM
- Skenario Umum untuk Peran: Pengguna, Aplikasi, dan Layanan

Pertimbangkan untuk mengenkripsi data diam

Anda memiliki opsi berikut untuk melindungi data diam di Amazon S3:

- Enkripsi sisi server — Semua bucket Amazon S3 memiliki enkripsi yang dikonfigurasi secara default, dan semua objek baru yang diunggah ke bucket S3 secara otomatis dienkripsi saat istirahat. Enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) adalah konfigurasi enkripsi default untuk setiap bucket di Amazon S3. Untuk menggunakan jenis enkripsi yang

berbeda, Anda dapat menentukan jenis enkripsi di sisi server yang akan digunakan dalam permintaan PUT S3, atau Anda dapat mengatur konfigurasi enkripsi default di bucket tujuan.

Amazon S3 juga menyediakan opsi enkripsi sisi server ini:

- Enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS)
- Enkripsi sisi server dua lapis dengan kunci () (DSSE-KMS AWS Key Management Service)AWS KMS
- Enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C)

Untuk informasi selengkapnya, lihat Melindungi data dengan enkripsi di sisi klien.

- Enkripsi di sisi klien–Enkripsikan data sisi klien dan unggah data yang dienkripsi ke Amazon S3. Dalam hal ini, Anda mengelola proses enkripsi, kunci enkripsi, dan alat terkait. Seperti enkripsi di sisi server, enkripsi di sisi klien dapat membantu mengurangi risiko dengan mengenkripsi data dengan kunci yang disimpan dalam mekanisme yang berbeda dari mekanisme yang menyimpan data itu sendiri.

  Amazon S3 menyediakan beberapa opsi enkripsi di sisi klien. Untuk informasi selengkapnya, lihat Melindungi data menggunakan enkripsi di sisi klien.

Menerapkan enkripsi data bergerak

Anda dapat menggunakan HTTPS (TLS) untuk membantu mencegah penyerang potensial menguping atau memanipulasi lalu lintas jaringan dengan menggunakan atau serangan serupa. person-in-the-middle Sebaiknya izinkan koneksi terenkripsi melalui HTTPS (TLS) dengan menggunakan kondisi aws:SecureTransport dalam kebijakan bucket Amazon S3 Anda.

> **Important**
>
> Kami menyarankan agar aplikasi Anda tidak menyematkan sertifikat Amazon S3 TLS karena AWS tidak mendukung penyematan sertifikat tepercaya publik. S3 secara otomatis memperbarui sertifikat dan perpanjangan dapat terjadi kapan saja sebelum sertifikat kedaluwarsa. Memperbarui sertifikat menghasilkan key pair public-private baru. Jika Anda telah menyematkan sertifikat S3 yang baru saja diperbarui dengan kunci publik baru, Anda tidak akan dapat terhubung ke S3 hingga aplikasi Anda menggunakan sertifikat baru.

Lalu, pertimbangkan untuk menerapkan kontrol detektif yang sedang berlangsung dengan menggunakan aturan s3-bucket-ssl-requests-only AWS Config terkelola.

Pertimbangkan untuk menggunakan Kunci Objek S3

Dengan Kunci Objek S3, Anda dapat menyimpan objek dengan menggunakan model “Tulis Sekali Baca Banyak” (WORM). Kunci Objek S3 dapat membantu mencegah penghapusan data yang tidak disengaja atau tidak tepat. Misalnya, Anda dapat menggunakan S3 Object Lock untuk membantu melindungi AWS CloudTrail log Anda.

Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek S3.

Aktifkan Penentuan Versi S3

Penentuan Versi S3 adalah cara menyimpan beberapa varian objek dalam bucket yang sama. Anda dapat menggunakan Penentuan Versi untuk menyimpan, mengambil, dan memulihkan setiap versi dari setiap objek yang disimpan dalam bucket Anda. Dengan Penentuan Versi, Anda dapat dengan mudah memulihkan dari tindakan pengguna yang tidak diinginkan, serta kegagalan aplikasi.

Lalu, pertimbangkan untuk menerapkan kontrol detektif yang sedang berlangsung dengan menggunakan aturan s3-bucket-versioning-enabled AWS Config terkelola.

Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.

Pertimbangkan untuk menggunakan Replikasi Lintas Wilayah S3

Meskipun Amazon S3 menyimpan data Anda di berbagai Zona Ketersediaan yang berbeda secara geografis secara default, persyaratan kepatuhan mungkin mengatur bahwa Anda menyimpan data dengan jarak lebih jauh. Dengan S3 Cross-Region Replication (CRR), Anda dapat mereplikasi data antara jarak jauh Wilayah AWS untuk membantu memenuhi persyaratan ini. CRR memungkinkan penyalinan objek secara otomatis dan asinkron di seluruh ember dalam berbagai jenis. Wilayah AWS Untuk informasi selengkapnya, lihat Mereplikasi ikhtisar objek.

> **Note**
>
> CRR mewajibkan bucket S3 sumber dan tujuan agar Penentuan Versi diaktifkan.

Lalu, pertimbangkan untuk menerapkan kontrol detektif yang sedang berlangsung dengan menggunakan aturan s3-bucket-replication-enabled AWS Config terkelola.

Pertimbangkan untuk menggunakan titik akhir VPC untuk akses Amazon S3

Titik akhir cloud privat virtual (VPC) untuk Amazon S3 adalah entitas logis dalam sebuah VPC yang mengizinkan konektivitas hanya ke Amazon S3. Titik akhir VPC dapat membantu mencegah lalu lintas melintasi internet terbuka.

Titik akhir VPC untuk Amazon S3 menyediakan banyak cara untuk mengontrol akses ke data Amazon S3 Anda:

- Anda dapat mengontrol permintaan, pengguna, atau grup yang diizinkan melalui titik akhir VPC tertentu dengan menggunakan kebijakan bucket S3.
- Anda dapat mengendalikan VPC atau titik akhir VPC mana yang memiliki akses ke bucket S3 Anda dengan menggunakan kebijakan bucket S3.
- Anda dapat membantu mencegah eksfiltrasi data dengan menggunakan VPC yang tidak memiliki gateway internet.

Untuk informasi selengkapnya, lihat Megendalikan akses dari titik akhir VPC dengan kebijakan bucket.

Menggunakan layanan AWS keamanan terkelola untuk memantau keamanan data

Beberapa layanan AWS keamanan terkelola dapat membantu Anda mengidentifikasi, menilai, dan memantau risiko keamanan dan kepatuhan untuk data Amazon S3 Anda. Layanan ini juga dapat membantu Anda melindungi data Anda dari risiko tersebut. Layanan ini mencakup kemampuan deteksi, pemantauan, dan perlindungan otomatis yang dirancang untuk menskalakan dari sumber daya Amazon S3 untuk sumber daya tunggal Akun AWS untuk organisasi yang mencakup ribuan akun.

Untuk informasi selengkapnya, lihat Memantau keamanan data dengan layanan AWS keamanan terkelola.

Praktik Terbaik Pemantauan dan Audit Amazon S3

Praktik terbaik berikut untuk Amazon S3 dapat membantu mendeteksi potensi kelemahan dan insiden keamanan.

Identifikasi dan audit semua bucket Amazon S3 Anda

Identifikasi aset IT Anda adalah aspek penting dari tata kelola dan keamanan. Anda perlu memiliki visibilitas semua sumber daya Amazon S3 Anda untuk menilai postur keamanan dan mengambil

tindakan di area kelemahan potensial. Untuk mengaudit sumber daya Anda, kami sarankan melakukan hal berikut ini:

- Gunakan Editor Tag untuk mengidentifikasi dan menandai sumber daya yang sensitif terhadap keamanan atau audit, kemudian gunakan tag tersebut saat Anda perlu mencari sumber daya ini. Untuk informasi selengkapnya, lihat Mencari Sumber Daya untuk Ditandai di Panduan Pengguna AWS Sumber Daya Penandaan.
- Gunakan Inventaris S3 untuk mengaudit dan melaporkan replikasi dan status enkripsi objek Anda untuk kebutuhan bisnis, kepatuhan, dan regulasi. Untuk informasi selengkapnya, lihat Inventaris Amazon S3.
- Buat grup sumber daya untuk sumber daya Amazon S3 Anda. Untuk informasi selengkapnya, lihat Apa yang dimaksud dengan grup sumber daya? dalam AWS Resource Groups Panduan Pengguna.

Melaksanakan pemantauan dengan menggunakan alat AWS pemantauan

Pemantauan adalah bagian penting dalam menjaga keandalan, keamanan, ketersediaan, dan kinerja Amazon S3 dan solusi Anda AWS . AWS menyediakan beberapa alat dan layanan untuk membantu Anda memantau Amazon S3 dan yang lain. Layanan AWS Misalnya, Anda dapat memantau CloudWatch metrik Amazon untuk Amazon S3, khususnya,, `PutRequests` dan `GetRequests` metrik. `4xxErrors` `DeleteRequests` Untuk informasi selengkapnya, lihat Memantau metrik dengan Amazon CloudWatch dan Pemantauan Amazon S3.

Untuk contoh kedua, lihat Contoh: Aktivitas Bucket Amazon S3. Contoh ini menjelaskan cara membuat CloudWatch alarm yang dipicu saat panggilan API Amazon S3 dibuat `PUT` atau `DELETE` kebijakan bucket, siklus hidup bucket, atau konfigurasi replikasi bucket, atau ke ACL bucket. `PUT`

Aktifkan log akses server Amazon S3

Pencatatan akses server memberikan catatan detail permintaan yang dibuat ke bucket. Log akses server dapat membantu Anda dalam keamanan dan audit akses, membantu Anda mempelajari basis pelanggan Anda, dan memahami tagihan Amazon S3. Untuk petunjuk tentang mengaktifkan log akses server, lihat Pencatatan permintaan dengan pencatatan akses server.

Juga pertimbangkan untuk menerapkan kontrol detektif yang sedang berlangsung dengan menggunakan aturan s3-bucket-logging-enabled AWS Config terkelola.

Gunakan AWS CloudTrail

AWS CloudTrail menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau Layanan AWS di Amazon S3. Anda dapat menggunakan informasi yang dikumpulkan oleh CloudTrail untuk menentukan hal-hal berikut:

- Permintaan yang diajukan ke Amazon S3
- Alamat IP dari mana permintaan itu dibuat
- Siapa yang membuat permintaan
- Kapan permintaan dibuat
- Detail tambahan tentang permintaan

Misalnya, Anda dapat mengidentifikasi CloudTrail entri untuk PUT tindakan yang memengaruhi akses data, khususnya`PutBucketAcl`, `PutObjectAclPutBucketPolicy`, dan`PutBucketWebsite`.

Saat Anda mengatur Akun AWS, CloudTrail diaktifkan secara default. Anda dapat melihat peristiwa terbaru di CloudTrail konsol. Untuk membuat catatan aktivitas dan peristiwa yang sedang berlangsung untuk bucket Amazon S3, Anda dapat membuat jejak di konsol. CloudTrail Untuk informasi selengkapnya, lihat Mencatat peristiwa data dalam AWS CloudTrail Panduan Pengguna.

Saat membuat jejak, Anda dapat mengonfigurasi CloudTrail untuk mencatat peristiwa data. Peristiwa data adalah catatan operasi sumber daya yang dilakukan pada atau di dalam sumber daya. Di Amazon S3, peristiwa data merekam aktivitas API tingkat objek untuk masing-masing bucket. CloudTrail mendukung subset operasi API tingkat objek Amazon S3, sepertiGetObject,, dan. `DeleteObject` `PutObject` Untuk informasi selengkapnya tentang cara CloudTrail bekerja dengan Amazon S3, lihat. Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail Di konsol Amazon S3, Anda juga dapat mengonfigurasi bucket S3 Anda ke Mengaktifkan pencatatan CloudTrail peristiwa untuk bucket dan objek S3.

AWS Config menyediakan aturan terkelola (`cloudtrail-s3-dataevents-enabled`) yang dapat Anda gunakan untuk mengonfirmasi bahwa setidaknya satu CloudTrail jejak mencatat peristiwa data untuk bucket S3 Anda. Untuk informasi lebih lanjut, lihat cloudtrail-s3-dataevents-enabled dalam Panduan Pengembang AWS Config .

Aktifkan AWS Config

Beberapa praktik terbaik yang tercantum dalam topik ini menyarankan untuk membuat AWS Config aturan. AWS Config membantu Anda menilai, mengaudit, dan mengevaluasi konfigurasi AWS sumber daya Anda. AWS Config memantau konfigurasi sumber daya sehingga Anda dapat mengevaluasi konfigurasi yang direkam terhadap konfigurasi aman yang diinginkan. Dengan AWS Config, Anda dapat melakukan hal berikut:

- Tinjau perubahan konfigurasi dan hubungan antar sumber daya AWS

- Selidiki riwayat konfigurasi sumber daya terperinci
- Tentukan kepatuhan Anda secara keseluruhan terhadap konfigurasi yang ditentukan dalam pedoman internal Anda

Menggunakan AWS Config dapat membantu Anda menyederhanakan audit kepatuhan, analisis keamanan, manajemen perubahan, dan pemecahan masalah operasional. Untuk informasi selengkapnya, lihat Menyiapkan AWS Config dengan Konsol di Panduan AWS Config Pengembang. Saat menentukan jenis sumber daya yang akan dicatat, pastikan Anda menyertakan sumber daya Amazon S3.

> **Important**
>
> AWS Config aturan terkelola hanya mendukung bucket tujuan umum saat mengevaluasi sumber daya Amazon S3. AWS Config tidak merekam perubahan konfigurasi untuk bucket direktori. Untuk informasi selengkapnya, lihat Aturan AWS Config Terkelola dan Daftar Aturan AWS Config Terkelola di Panduan AWS Config Pengembang.

Untuk contoh cara menggunakan AWS Config, lihat Cara Menggunakan AWS Config untuk Memantau dan Menanggapi Bucket Amazon S3 yang Mengizinkan Akses Publik di Blog Keamanan.AWS

Temukan data sensitif dengan menggunakan Amazon Macie

Amazon Macie adalah layanan keamanan yang menemukan data sensitif dengan menggunakan machine learning dan pencocokan pola. Macie memberikan visibilitas ke dalam risiko keamanan data, dan memungkinkan perlindungan otomatis terhadap risiko tersebut. Dengan Macie, Anda dapat mengotomatisasi penemuan dan pelaporan data sensitif di properti data Amazon S3 untuk mendapatkan pemahaman yang lebih baik tentang data yang disimpan organisasi Anda di S3.

Untuk mendeteksi data sensitif dengan Macie, Anda dapat menggunakan kriteria dan teknik bawaan yang dirancang untuk mendeteksi daftar jenis data sensitif yang besar dan berkembang untuk banyak negara dan wilayah. Jenis data sensitif ini mencakup beberapa jenis informasi pengenal pribadi (PII), data keuangan, dan data kredensial. Anda juga dapat menggunakan kriteria kustom yang Anda tetapkan, ekspresi reguler yang menentukan pola teks yang cocok dan, opsional, urutan karakter dan aturan kedekatan yang menyempurnakan hasil.

Jika Macie mendeteksi data sensitif dalam S3 Object, Macie akan membuat temuan keamanan untuk memberi tahu Anda. Temuan ini memberikan informasi tentang objek yang terpengaruh,

jenis dan jumlah kemunculan data sensitif yang ditemukan Macie, dan detail tambahan untuk membantu Anda menyelidiki bucket dan S3 Object yang terpengaruh. Untuk informasi selengkapnya, lihat Panduan Pengguna Amazon Macie.

Menggunakan Lensa Penyimpanan S3

Lensa Penyimpanan S3 adalah fitur analisis penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan objek. Lensa Penyimpanan S3 juga menganalisis metrik untuk memberikan rekomendasi kontekstual yang dapat Anda gunakan untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik untuk melindungi data Anda.

Dengan Lensa Penyimpanan S3, Anda dapat menggunakan metrik untuk menghasilkan wawasan ringkasan, seperti mencari tahu berapa banyak penyimpanan yang Anda miliki di seluruh organisasi atau bucket dan prefiks mana yang paling cepat berkembang. Anda juga dapat menggunakan metrik Lensa Penyimpanan S3 untuk mengidentifikasi peluang optimasi biaya, menerapkan praktik terbaik untuk perlindungan data dan keamanan, serta meningkatkan kinerja beban kerja aplikasi.

Misalnya, Anda dapat mengidentifikasi bucket yang tidak memiliki aturan Siklus Hidup S3 untuk membatalkan unggahan multibagian yang tidak lengkap yang berusia lebih dari 7 hari. Anda juga dapat mengidentifikasi bucket yang tidak mengikuti praktik terbaik perlindungan data, seperti menggunakan Replikasi S3 atau Penentuan Versi S3. Untuk informasi selengkapnya, lihat Memahami Lensa Penyimpanan Amazon S3.

Memantau laporan keamanan AWS

Kami menyarankan Anda secara teratur memeriksa nasihat keamanan yang diposting dalam Trusted Advisor untuk Akun AWS Anda. Secara khusus, cari peringatan tentang bucket Amazon S3 dengan “buka izin akses.” Anda dapat melakukan ini secara terprogram dengan menggunakan describe-trusted-advisor-checks.

Selanjutnya, secara aktif memantau alamat email utama yang terdaftar untuk masing-masing Anda Akun AWS. AWS menggunakan alamat email ini untuk menghubungi Anda tentang masalah keamanan yang muncul yang mungkin memengaruhi Anda.

AWS masalah operasional dengan dampak luas diposting di AWS Health Dashboard - Layanan kesehatan. Masalah operasional juga diposting ke akun individu melalui AWS Health Dashboard. Untuk informasi lebih lanjut, lihat dokumentasi AWS Health.

Memantau keamanan data dengan layanan AWS keamanan terkelola

Beberapa layanan AWS keamanan terkelola dapat membantu Anda mengidentifikasi, menilai, dan memantau risiko keamanan dan kepatuhan untuk data Amazon S3 Anda. Mereka juga dapat membantu Anda melindungi data Anda dari risiko tersebut. Layanan ini mencakup kemampuan deteksi, pemantauan, dan perlindungan otomatis yang dirancang untuk menskalakan dari sumber daya Amazon S3 untuk sumber daya tunggal Akun AWS hingga sumber daya untuk organisasi yang mencakup ribuan. Akun AWS

AWS Layanan deteksi dan respons dapat membantu Anda mengidentifikasi potensi kesalahan konfigurasi keamanan, ancaman, atau perilaku tak terduga, sehingga Anda dapat dengan cepat merespons aktivitas yang berpotensi tidak sah atau berbahaya di lingkungan Anda. AWS Layanan perlindungan data dapat membantu Anda memantau dan melindungi data, akun, dan beban kerja Anda dari akses yang tidak sah. Mereka juga dapat membantu Anda menemukan data sensitif, seperti informasi pengenal pribadi (PII), di properti data Amazon S3 Anda.

Untuk membantu Anda mengidentifikasi dan mengevaluasi risiko keamanan dan kepatuhan data, layanan keamanan terkelola AWS menghasilkan temuan untuk memberi tahu Anda tentang potensi peristiwa keamanan atau masalah dengan data Amazon S3 Anda. Temuan ini memberikan rincian relevan yang dapat Anda gunakan untuk menyelidiki, menilai, dan menindaklanjuti risiko ini sesuai dengan alur kerja dan kebijakan respons insiden Anda. Anda dapat mengakses data temuan secara langsung dengan menggunakan setiap layanan. Anda juga dapat mengirim data ke aplikasi, layanan, dan sistem lain, seperti insiden keamanan dan sistem manajemen peristiwa (SIEM).

Untuk memantau keamanan data Amazon S3 Anda, pertimbangkan untuk menggunakan layanan AWS keamanan terkelola ini.

Amazon GuardDuty

Amazon GuardDuty adalah layanan pendeteksi ancaman yang terus memantau aktivitas berbahaya dan beban kerja Anda Akun AWS dan memberikan temuan keamanan terperinci untuk visibilitas dan remediasi.

Dengan fitur perlindungan S3 di GuardDuty, Anda dapat mengonfigurasi GuardDuty untuk menganalisis AWS CloudTrail manajemen dan peristiwa data untuk sumber daya Amazon S3 Anda. GuardDuty kemudian memantau peristiwa tersebut untuk aktivitas berbahaya dan mencurigakan. Untuk menginformasikan analisis dan mengidentifikasi potensi risiko keamanan, GuardDuty gunakan umpan intelijen ancaman dan pembelajaran mesin.

GuardDuty dapat memantau berbagai jenis aktivitas untuk sumber daya Amazon S3 Anda. Misalnya, peristiwa CloudTrail manajemen untuk Amazon S3 mencakup operasi tingkat ember, seperti,, dan. `ListBuckets` `DeleteBucket` `PutBucketReplication` CloudTrail peristiwa data untuk Amazon S3 mencakup operasi tingkat objek, seperti`GetObject`,, dan. `ListObjects` `PutObject` Jika GuardDuty mendeteksi aktivitas anomali atau berpotensi berbahaya, itu menghasilkan temuan untuk memberi tahu Anda.

Untuk informasi selengkapnya, lihat Perlindungan Amazon S3 GuardDuty di Amazon di GuardDuty Panduan Pengguna Amazon.

Amazon Detective

Amazon Detective menyederhanakan proses investigasi dan membantu Anda melakukan investigasi keamanan yang lebih cepat dan efektif. Detective menyediakan agregasi data, ringkasan, dan konteks yang dapat membantu Anda menganalisis dan menilai sifat dan tingkat kemungkinan masalah keamanan.

Detective secara otomatis mengekstrak peristiwa berbasis waktu, seperti panggilan API dari dan AWS CloudTrail Amazon VPC Flow Logs untuk sumber daya Anda. AWS Ini juga menelan temuan yang dihasilkan oleh Amazon GuardDuty. Detective kemudian menggunakan machine learning, analisis statistik, dan teori grafik untuk menghasilkan visualisasi yang membantu Anda melakukan investigasi keamanan yang efektif dengan lebih cepat.

Visualisasi ini memberikan pandangan interaktif yang terpadu tentang perilaku sumber daya dan interaksi di antara mereka dari waktu ke waktu. Anda dapat menjelajahi grafik perilaku ini untuk memeriksa tindakan yang berpotensi berbahaya, seperti upaya login yang gagal atau panggilan API yang mencurigakan. Anda juga dapat melihat bagaimana tindakan ini memengaruhi sumber daya, seperti bucket dan S3 Object.

Untuk informasi selengkapnya, lihat Panduan Administrasi Amazon Detective.

IAM Access Analyzer

AWS Identity and Access Management Access Analyzer (IAM Access Analyzer) dapat membantu Anda mengidentifikasi sumber daya yang dibagikan dengan entitas eksternal. Anda juga dapat menggunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM terhadap tata bahasa kebijakan dan praktik terbaik, serta menghasilkan kebijakan IAM berdasarkan aktivitas akses di log Anda. AWS CloudTrail

IAM Access Analyzer menggunakan penalaran berbasis logika untuk menganalisis kebijakan sumber daya di AWS lingkungan Anda, seperti kebijakan bucket. Dengan IAM Access Analyzer

untuk S3, Anda akan diberi tahu saat bucket S3 dikonfigurasi untuk memungkinkan akses ke siapa pun di internet atau lainnya Akun AWS, termasuk akun di luar organisasi Anda. Misalnya, Penganalisis Akses IAM untuk S3 dapat melaporkan bahwa suatu bucket telah membaca atau menulis akses yang disediakan melalui daftar kontrol akses (ACL) bucket, kebijakan bucket, kebijakan titik akses Multi-Wilayah atau kebijakan titik akses. Untuk setiap bucket publik atau bucket bersama, Anda akan menerima temuan yang menunjukkan sumber dan tingkat akses publik atau bersama. Dengan temuan ini, Anda dapat segera mengambil tindakan korektif yang tepat untuk memulihkan akses bucket ke apa yang Anda inginkan.

Untuk informasi selengkapnya, lihat Meninjau akses bucket menggunakan IAM Access Analyzer untuk S3.

Amazon Macie

Amazon Macie adalah layanan keamanan data yang menemukan data sensitif dengan menggunakan machine learning dan pencocokan pola, memberikan visibilitas ke risiko keamanan data, dan memungkinkan perlindungan otomatis terhadap risiko tersebut.

Dengan Macie, Anda dapat mengotomatisasi penemuan dan pelaporan data sensitif di bucket S3 Anda untuk mendapatkan pemahaman yang lebih baik tentang data yang disimpan organisasi Anda di Amazon S3. Untuk mendeteksi data sensitif, Anda dapat menggunakan kriteria dan teknik bawaan yang disediakan Macie, kriteria kustom yang Anda tentukan, atau kombinasi keduanya. Jika Macie mendeteksi data sensitif dalam S3 Object, Macie akan membuat temuan untuk memberi tahu Anda. Temuan ini memberikan informasi tentang bucket dan objek yang terpengaruh, jenis dan jumlah peristiwa data sensitif yang ditemukan Macie, dan detail tambahan untuk memfasilitasi penyelidikan Anda.

Macie juga menyediakan statistik dan data lainnya yang menawarkan wawasan tentang postur keamanan data Amazon S3 Anda, dan secara otomatis mengevaluasi dan memantau bucket S3 Anda untuk keamanan dan kontrol akses. Jika Macie mendeteksi potensi masalah dengan keamanan atau privasi data Anda, seperti bucket yang dapat diakses publik, Macie akan membuat temuan untuk Anda tinjau dan perbaiki seperlunya.

Untuk informasi selengkapnya, lihat Panduan Pengguna Amazon Macie.

AWS Security Hub

AWS Security Hub adalah layanan manajemen postur keamanan yang melakukan pemeriksaan praktik terbaik keamanan, mengumpulkan peringatan dan temuan dari berbagai sumber ke dalam satu format, dan memungkinkan remediasi otomatis.

Security Hub mengumpulkan dan menyediakan data temuan keamanan dari solusi AWS Partner Network keamanan terintegrasi dan Layanan AWS, termasuk Amazon Detective, Amazon, IAM Access Analyzer, dan GuardDuty Amazon Macie. Ini juga menghasilkan temuannya sendiri dengan menjalankan pemeriksaan keamanan otomatis yang berkelanjutan berdasarkan praktik AWS terbaik dan standar industri yang didukung.

Security Hub kemudian mengkorelasikan dan mengkonsolidasikan temuan di seluruh penyedia untuk membantu Anda memprioritaskan dan memproses temuan yang paling signifikan. Ini juga menyediakan dukungan untuk tindakan kustom, yang dapat Anda gunakan untuk menginvokasi respons atau tindakan remediasi untuk kelas temuan tertentu.

Dengan Security Hub, Anda dapat menilai status keamanan dan kepatuhan sumber daya Amazon S3, dan Anda dapat melakukannya sebagai bagian dari analisis yang lebih luas tentang postur keamanan organisasi Anda di masing-masing Wilayah AWS dan di beberapa Wilayah. Ini termasuk menganalisis tren keamanan dan mengidentifikasi masalah keamanan prioritas tertinggi. Anda juga dapat mengumpulkan temuan dari beberapa Wilayah AWS, dan memantau serta memproses data temuan agregat dari satu Wilayah.

Untuk informasi selengkapnya, lihat Kontrol Amazon Simple Storage Service di dalam Panduan Pengguna AWS Security Hub .

Mengelola penyimpanan Amazon S3 Anda

Setelah membuat bucket dan mengunggah objek di Amazon S3, Anda dapat mengelola penyimpanan objek menggunakan fitur seperti Penentuan Versi, kelas penyimpanan, penguncian objek, operasi batch, replikasi, tag, dan banyak lagi. Bagian berikut memberikan informasi terperinci tentang kemampuan pengelolaan penyimpanan dan fitur yang tersedia di Amazon S3.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Topik

- Menggunakan Penentuan Versi dalam bucket S3
- MenggunakanAWS Backup untuk Amazon S3
- Bekerja dengan objek yang diarsipkan
- Menggunakan Kunci Objek S3
- Menggunakan kelas penyimpanan Amazon S3
- Penyimpanan data jangka panjang menggunakan kelas penyimpanan S3 Glacier
- Amazon S3 Intelligent-Tiering
- Mengelola siklus hidup penyimpanan Anda
- Inventaris Amazon S3
- Mereplikasi ikhtisar objek
- Mengategorikan penyimpanan Anda menggunakan tag
- Menggunakan tag alokasi biaya bucket S3
- Pelaporan penagihan dan penggunaan untuk Amazon S3
- Menyaring dan mengambil data menggunakan Amazon S3 Select
- Melakukan operasi batch berskala besar pada objek Amazon S3

Menggunakan Penentuan Versi dalam bucket S3

Penentuan Versi di Amazon S3 adalah cara menyimpan beberapa varian objek dalam bucket yang sama. Anda dapat menggunakan fitur Penentuan Versi S3 untuk menyimpan, mengambil, dan memulihkan setiap versi dari setiap objek yang disimpan dalam bucket Anda. Dengan Penentuan Versi, Anda dapat lebih mudah memulihkan dari tindakan pengguna yang tidak diinginkan dan kegagalan aplikasi. Setelah Penentuan Versi diaktifkan untuk bucket, jika Amazon S3 menerima beberapa permintaan tulis untuk objek yang sama secara bersamaan, itu akan menyimpan semua objek.

Bucket dengan Penentuan Versi yang diaktifkan memungkinkan Anda memulihkan objek dari penghapusan atau penimpaan yang tidak disengaja. Misalnya, jika Anda menghapus objek, Amazon S3 menyisipkan penanda hapus alih-alih menghapus objek secara permanen. Penanda hapus menjadi versi objek saat ini. Jika Anda menimpa sebuah objek, akan muncul versi objek baru dalam bucket. Anda dapat selalu memulihkan versi sebelumnya. Untuk informasi selengkapnya, lihat Menghapus versi objek dari bucket dengan dukungan Penentuan Versi.

Secara default, Penentuan Versi S3 dinonaktifkan pada bucket, dan Anda harus mengaktifkannya secara eksplisit. Untuk informasi selengkapnya, lihat Mengaktifkan Penentuan Versi pada bucket.

> **Note**
>
> - SOAP API tidak mendukung Penentuan Versi S3. Dukungan SOAP melalui HTTP dihilangkan, tetapi masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP.
> - Tarif normal Amazon S3 berlaku untuk setiap versi objek yang disimpan dan ditransfer. Setiap versi objek adalah seluruh objek; bukan hanya sebuah diff dari versi sebelumnya. Dengan demikian, jika memiliki tiga versi objek yang disimpan, Anda akan dikenakan biaya untuk tiga objek.

Bucket tanpa versi, dengan dukungan Penentuan Versi, dan dengan Penentuan Versi ditangguhkan

Bucket bisa berada di salah satu dari tiga status:

- Tanpa versi (default)

- Dengan dukungan Penentuan Versi
- Dengan Penentuan Versi ditangguhkan

Anda mengaktifkan dan menangguhkan Penentuan Versi di tingkat bucket. Setelah Anda mengaktifkan Penentuan Versi sebuah bucket, itu tidak akan pernah dapat kembali ke status tanpa versi. Namun, Anda dapat menangguhkan Penentuan Versi pada bucket tersebut.

Status Penentuan Versi berlaku untuk semua (tidak pernah sebagian) objek dalam bucket tersebut. Saat Anda mengaktifkan Penentuan Versi di bucket, semua objek baru akan mendapatkan Penentuan Versi dan diberi ID versi unik. Objek yang sudah ada di bucket pada saat Penentuan Versi diaktifkan akan selalu mendapatkan Penentuan Versi dan diberi ID versi unik saat diubah oleh permintaan di masa mendatang. Perhatikan hal-hal berikut:

- Objek yang disimpan di bucket sebelum Anda mengatur status Penentuan Versi memiliki ID versi `null`. Saat Anda mengaktifkan Penentuan Versi, objek yang ada di bucket tidak berubah. Perubahannya adalah cara Amazon S3 menangani objek di masa mendatang. Untuk informasi selengkapnya, lihat Bekerja dengan objek di dalam bucket dengan dukungan Penentuan Versi.
- Pemilik bucket (atau pengguna dengan izin yang sesuai) dapat menangguhkan Penentuan Versi untuk menghentikan pengumpulan versi objek. Saat Anda menangguhkan Penentuan Versi, objek yang ada di bucket tidak berubah. Perubahannya adalah cara Amazon S3 menangani objek di masa mendatang. Untuk informasi selengkapnya, lihat Bekerja dengan objek dalam bucket dengan Penentuan Versi ditangguhkan.

Menggunakan Penentuan Versi S3 dengan Siklus Hidup S3

Untuk menyesuaikan pendekatan retensi data Anda dan mengontrol biaya penyimpanan, gunakan Penentuan Versi objek dengan Siklus Hidup S3. Untuk informasi selengkapnya, lihat Mengelola siklus hidup penyimpanan Anda. Untuk informasi tentang membuat konfigurasi Siklus Hidup S3 menggunakan,, AWS SDK AWS Management Console AWS CLI, atau REST API, lihat. Menyetel konfigurasi siklus hidup pada bucket

> **Important**
>
> Jika Anda memiliki konfigurasi siklus hidup kedaluwarsa objek di bucket tanpa versi dan ingin mempertahankan perilaku penghapusan permanen yang sama saat mengaktifkan Penentuan Versi, Anda harus menambahkan konfigurasi kedaluwarsa yang lama. Konfigurasi siklus aktif kedaluwarsa nonterkini mengelola penghapusan versi objek nonterkini di bucket

dengan dukungan Penentuan Versi. (Bucket dengan dukungan Penentuan Versi akan mempertahankan satu objek versi terkini dan nol atau beberapa versi lama.) Untuk informasi selengkapnya, lihat Menyetel konfigurasi siklus hidup pada bucket.

Untuk informasi tentang bekerja dengan Penentuan Versi S3, lihat topik-topik berikut.

Topik

- Cara kerja Penentuan Versi S3
- Mengaktifkan Penentuan Versi pada bucket
- Mengonfigurasi penghapusan MFA
- Bekerja dengan objek di dalam bucket dengan dukungan Penentuan Versi
- Bekerja dengan objek dalam bucket dengan Penentuan Versi ditangguhkan

Cara kerja Penentuan Versi S3

Gunakan Penentuan Versi S3 untuk menyimpan beberapa versi dari sebuah objek di dalam satu bucket sehingga Anda dapat memulihkan objek yang terhapus atau ditimpa secara tidak sengaja. Misalnya, jika Anda menerapkan Penentuan Versi S3 ke bucket, perubahan berikut akan terjadi:

- Jika Anda menghapus objek, alih-alih menghapus objek secara permanen, Amazon S3 menyisipkan penanda hapus yang menjadi versi objek saat ini. Anda dapat selalu memulihkan versi sebelumnya. Untuk informasi selengkapnya, lihat Menghapus versi objek dari bucket dengan dukungan Penentuan Versi.
- Jika Anda menimpa sebuah objek, Amazon S3 akan menambahkan versi objek baru dalam bucket. Versi sebelumnya tetap ada di bucket dan menjadi versi lama. Anda dapat selalu memulihkan versi sebelumnya.

> **Note**
>
> Tarif normal Amazon S3 berlaku untuk setiap versi objek yang disimpan dan ditransfer. Setiap versi objek adalah seluruh objek; bukan hanya sebuah diff dari versi sebelumnya. Dengan demikian, jika memiliki tiga versi objek yang disimpan, Anda akan dikenakan biaya untuk tiga objek.

Setiap bucket S3 yang Anda buat memiliki subsumber daya Penentuan Versi yang terkait dengannya. (Untuk informasi selengkapnya, lihat Opsi konfigurasi bucket.) Secara default, bucket Anda tanpa versi, dan subsumber daya Penentuan Versi menyimpan konfigurasi Penentuan Versi yang kosong, sebagai berikut.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
</VersioningConfiguration>
```

Untuk mengaktifkan Penentuan Versi, Anda dapat mengirimkan permintaan ke Amazon S3 dengan konfigurasi Penentuan Versi yang menyertakan status `Enabled`.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Status>Enabled</Status>
</VersioningConfiguration>
```

Untuk menangguhkan Penentuan Versi, Anda menetapkan nilai status ke `Suspended`.

> **Note**
>
> Saat Anda mengaktifkan Penentuan Versi pada bucket untuk pertama kalinya, mungkin diperlukan beberaoa saat agar perubahan dapat dilakukan sepenuhnya. Kami menyarankan Anda menunggu selama 15 menit setelah mengaktifkan Penentuan Versi sebelum mengeluarkan operasi tulis (`PUT` atau `DELETE`) pada objek di dalam bucket.

Pemilik bucket dan semua pengguna resmi AWS Identity and Access Management (IAM) dapat mengaktifkan pembuatan versi. Pemilik ember adalah Akun AWS yang menciptakan ember. Untuk informasi selengkapnya tentang izin, lihat Identity and Access Management untuk Amazon S3.

Untuk informasi selengkapnya tentang mengaktifkan dan menonaktifkan Pembuatan Versi S3 dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau REST API, lihat. the section called “Mengaktifkan Penentuan Versi pada bucket”

Topik

- ID versi
- Alur kerja Penentuan Versi

ID versi

Jika Anda mengaktifkan Penentuan Versi untuk bucket, Amazon S3 secara otomatis menghasilkan ID versi unik untuk objek yang sedang disimpan. Misalnya, dalam satu bucket Anda dapat memiliki dua objek dengan kunci (nama objek) yang sama tetapi ID versi yang berbeda, seperti `photo.gif` (versi 111111) dan `photo.gif` (versi 121212).
Diagram yang menggambarkan bucket berkemampuan versi yang memiliki dua objek dengan kunci yang sama tetapi ID versi berbeda.

Setiap objek memiliki ID versi, terlepas dari apakah Penentuan Versi S3 diaktifkan atau tidak. Jika Penentuan Versi S3 tidak diaktifkan, Amazon S3 menetapkan nilai ID versi ke `null`. Jika Penentuan Versi S3 diaktifkan, Amazon S3 akan menetapkan nilai ID versi untuk objek tersebut. Nilai ini membedakan objek dengan versi lain dari kunci yang sama.

Saat Anda mengaktifkan Penentuan Versi S3 pada bucket yang sudah ada, objek yang sudah disimpan dalam bucket tidak akan berubah. ID versi-nya (`null`), konten, dan izin akan tetap sama. Setelah Anda mengaktifkan Penentuan Versi S3, setiap objek yang ditambahkan ke bucket akan mendapatkan sebuah ID versi, yang membedakannya dengan versi lain dari kunci yang sama.

Hanya Amazon S3 yang menghasilkan ID versi, dan tidak dapat diedit. ID versi adalah string Unicode, dikode UTF-8, siap untuk URL yang samar dengan ukuran tidak lebih dari 1.024 byte. Berikut adalah contohnya:

```
3sL4kqtJlcpXroDTDmJ+rmSpXd3dIbrHY+MTRCxf3vjVBH40Nr8X8gdRQBpUMLUo
```

> **Note**
>
> Untuk mempermudah, contoh lain dalam topik ini menggunakan ID yang jauh lebih pendek.

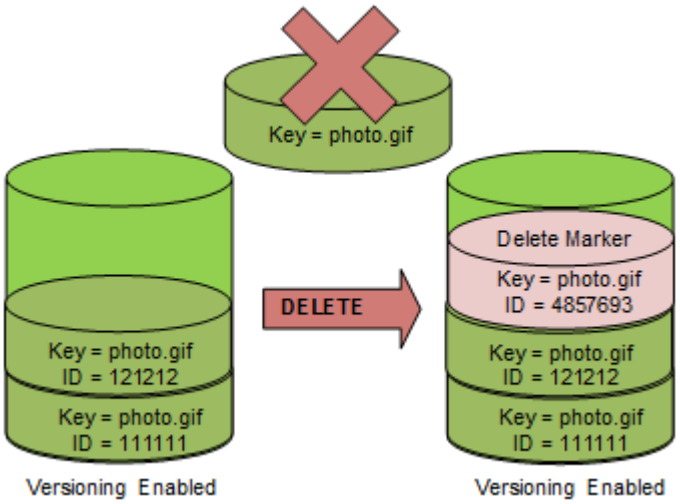
Alur kerja Penentuan Versi

Saat Anda `PUT` sebuah objek dalam sebuah bucket dengan dukungan Penentuan Versi, versi nonterkini tidak ditimpa. Seperti yang ditunjukkan pada gambar berikut, ketika versi `photo.gif` baru `PUT` ke dalam bucket yang sudah berisi objek dengan nama yang sama, perilaku berikut terjadi:

- Objek asli (ID = 111111) tetap berada di bucket.
- Amazon S3 menghasilkan ID versi baru (121212), dan menambahkan versi objek yang lebih baru ini ke bucket.

Dengan fungsi ini, Anda dapat mengambil versi objek sebelumnya jika objek telah ditimpa atau dihapus secara tidak sengaja.

Saat Anda `DELETE` sebuah objek, semua versi tetap berada dalam bucket, dan Amazon S3 menyisipkan penanda hapus, seperti yang ditunjukkan pada gambar berikut.



Delete marker menjadi versi objek saat ini. Secara default, permintaan `GET` akan mengambil versi yang paling terakhir disimpan. Melakukan permintaan `GET Object` ketika versi saat ini merupakan penanda hapus akan menampilkan pesan kesalahan `404 Not Found`, seperti yang ditunjukkan pada gambar berikut.

Akan tetapi, Anda dapat `GET` versi objek nonterkini dengan menentukan ID versinya. Di gambar berikut, Anda akan `GET` versi objek tertentu, 111111. Amazon S3 akan menampilkan versi objek tersebut meskipun bukan versi saat ini.

Untuk informasi selengkapnya, lihat Mengambil versi objek dari bucket dengan dukungan Penentuan Versi.

Anda dapat menghapus objek secara permanen dengan menentukan versi yang ingin Anda hapus. Hanya pemilik bucket Amazon S3 atau pengguna IAM resmi yang dapat menghapus versi secara permanen. Jika operasi `DELETE` Anda menentukan `versionId`, versi objek tersebut akan dihapus secara permanen, dan Amazon S3 tidak menyisipkan penanda hapus.

Anda dapat menambahkan lebih banyak keamanan dengan mengonfigurasi bucket untuk mengaktifkan penghapusan autentikasi multi-faktor (MFA). Saat Anda mengaktifkan penghapusan MFA untuk sebuah bucket, pemilik bucket harus menyertakan dua bentuk autentikasi dalam setiap permintaan untuk menghapus sebuah versi atau mengubah status Penentuan Versi bucket. Untuk informasi selengkapnya, lihat Mengonfigurasi penghapusan MFA.

Kapan versi baru dibuat untuk suatu objek?

Versi baru objek dibuat hanya ketika Anda PUT obyek baru. Ingatlah bahwa tindakan tertentu, seperti `CopyObject`, bekerja dengan menerapkan operasi PUT.

Beberapa tindakan yang mengubah objek saat ini tidak membuat versi baru karena tindakan tersebut tidak PUT objek baru. Ini termasuk tindakan seperti mengubah tag pada objek.

> **Important**
>
> Jika Anda menyadari adanya peningkatan signifikan dalam jumlah HTTP 503 (Layanan Tidak Tersedia) yang diterima oleh objek permintaan PUT atau DELETE Amazon S3 ke sebuah bucket dengan Penentuan Versi S3 yang diaktifkan, mungkin Anda memiliki satu atau beberapa objek dalam bucket yang memiliki jutaan versi. Untuk informasi selengkapnya, lihat bagian Penentuan Versi S3 dari Pemecahan Masalah.

Mengaktifkan Penentuan Versi pada bucket

Gunakan Penentuan Versi S3 untuk menyimpan beberapa versi dari sebuah objek di dalam satu bucket. Bagian ini memberikan contoh cara mengaktifkan pembuatan versi pada bucket menggunakan konsol, REST API, AWS SDK, dan AWS Command Line Interface ().AWS CLI

> **Note**
>
> Jika Anda mengaktifkan pembuatan versi pada bucket untuk pertama kalinya, mungkin diperlukan waktu hingga 15 menit agar perubahan disebarkan sepenuhnya. Kami menyarankan Anda menunggu selama 15 menit setelah mengaktifkan Penentuan Versi sebelum mengeluarkan operasi tulis (PUT atau DELETE) pada objek di dalam bucket. Operasi tulis yang dikeluarkan sebelum konversi ini selesai mungkin berlaku untuk objek yang tidak berversi.

Untuk informasi selengkapnya tentang Penentuan Versi S3, lihat Menggunakan Penentuan Versi dalam bucket S3. Untuk informasi tentang cara bekerja dengan objek yang ada di bucket dengan dukungan Penentuan Versi, lihat Bekerja dengan objek di dalam bucket dengan dukungan Penentuan Versi.

Untuk mempelajari lebih lanjut cara menggunakan Penentuan Versi S3 untuk melindungi data, lihat Tutorial: Melindungi data di Amazon S3 dari penghapusan yang tidak disengaja atau bug aplikasi menggunakan Penentuan Versi S3, Kunci Objek S3, dan Replikasi S3.

Setiap bucket S3 yang Anda buat memiliki subsumber daya Penentuan Versi yang terkait dengannya. (Untuk informasi selengkapnya, lihat Opsi konfigurasi bucket.) Secara default, bucket Anda tanpa versi, dan subsumber daya Penentuan Versi menyimpan konfigurasi Penentuan Versi yang kosong, sebagai berikut.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
</VersioningConfiguration>
```

Untuk mengaktifkan Penentuan Versi, Anda dapat mengirimkan permintaan ke Amazon S3 dengan konfigurasi Penentuan Versi yang menyertakan status.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Status>Enabled</Status>
</VersioningConfiguration>
```

Untuk menangguhkan Penentuan Versi, Anda menetapkan nilai status ke `Suspended`.

Pemilik bucket dan semua pengguna sah dapat mengaktifkan Penentuan Versi. Pemilik bucket adalah Akun AWS yang membuat bucket (akun root). Untuk informasi selengkapnya tentang izin, lihat Identity and Access Management untuk Amazon S3.

Bagian berikut memberikan detail lebih lanjut tentang mengaktifkan Versi S3 menggunakan konsol, AWS CLI, dan SDK. AWS

Menggunakan konsol S3

Ikuti langkah-langkah ini untuk menggunakan fitur AWS Management Console untuk mengaktifkan pembuatan versi pada bucket S3.

Untuk mengaktifkan atau menonaktifkan Penentuan Versi pada bucket S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang ingin Anda aktifkan Penentuan Versi-nya.
3. Pilih Properti.

4. Di bawah Penentuan Versi Bucket, pilih Edit.
5. Pilih Tangguhkan atau Aktifkan, lalu pilih Simpan perubahan.

> **Note**
>
> Anda dapat menggunakan otentikasi AWS multi-faktor (MFA) dengan pembuatan versi. Saat Anda menggunakan MFA dengan pembuatan versi, Anda harus memberikan kunci akses dan kode yang valid dari perangkat MFA akun untuk menghapus versi objek secara permanen atau menangguhkan atau mengaktifkan kembali pembuatan versi. Akun AWS Untuk menggunakan MFA dengan Penentuan Versi, Anda mengaktifkan `MFA Delete`. Namun, Anda tidak dapat mengaktifkan `MFA Delete` menggunakan AWS Management Console. Anda harus menggunakan AWS Command Line Interface (AWS CLI) atau API. Untuk informasi selengkapnya, lihat Mengonfigurasi penghapusan MFA.

Menggunakan AWS CLI

Contoh berikut mengaktifkan Penentuan Versi pada bucket S3.

```
aws s3api put-bucket-versioning --bucket example-s3-bucket1 --versioning-configuration
 Status=Enabled
```

Contoh berikut mengaktifkan penghapusan Penentuan Versi S3 dan autentikasi multi-faktor (MFA) pada bucket.

```
aws s3api put-bucket-versioning --bucket example-s3-bucket1 --versioning-configuration
 Status=Enabled,MFADelete=Enabled --mfa "SERIAL 123456"
```

> **Note**
>
> Menggunakan penghapusan MFA memerlukan perangkat autentikasi fisik atau virtual disetujui. Untuk informasi selengkapnya tentang menggunakan penghapusan MFA di Amazon S3, lihat Mengonfigurasi penghapusan MFA.

Untuk informasi selengkapnya tentang mengaktifkan versi menggunakan AWS CLI, lihat put-bucket-versioningdi Command Reference.AWS CLI

Menggunakan AWS SDK

Contoh berikut memungkinkan pembuatan versi pada bucket dan kemudian mengambil status pembuatan versi menggunakan dan. AWS SDK for Java AWS SDK for .NET Untuk informasi tentang penggunaan AWS SDK lainnya, lihat Pusat Developer AWS.

.NET

Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using System;
using Amazon.S3;
using Amazon.S3.Model;

namespace s3.amazon.com.docsamples
{
    class BucketVersioningConfiguration
    {
        static string bucketName = "*** bucket name ***";

        public static void Main(string[] args)
        {
            using (var client = new AmazonS3Client(Amazon.RegionEndpoint.USEast1))
            {
                try
                {
                    EnableVersioningOnBucket(client);
                    string bucketVersioningStatus =
 RetrieveBucketVersioningConfiguration(client);
                }
                catch (AmazonS3Exception amazonS3Exception)
                {
                    if (amazonS3Exception.ErrorCode != null &&
                        (amazonS3Exception.ErrorCode.Equals("InvalidAccessKeyId")
                        ||
                        amazonS3Exception.ErrorCode.Equals("InvalidSecurity")))
                    {
                        Console.WriteLine("Check the provided AWS Credentials.");
                        Console.WriteLine(
```

```
                        "To sign up for service, go to http://aws.amazon.com/s3");
                    }
                    else
                    {
                        Console.WriteLine(
                         "Error occurred. Message:'{0}' when listing objects",
                         amazonS3Exception.Message);
                    }
                }
            }

            Console.WriteLine("Press any key to continue...");
            Console.ReadKey();
        }

        static void EnableVersioningOnBucket(IAmazonS3 client)
        {

                PutBucketVersioningRequest request = new PutBucketVersioningRequest
                {
                    BucketName = bucketName,
                    VersioningConfig = new S3BucketVersioningConfig
                    {
                        Status = VersionStatus.Enabled
                    }
                };

                PutBucketVersioningResponse response =
client.PutBucketVersioning(request);
        }


        static string RetrieveBucketVersioningConfiguration(IAmazonS3 client)
        {
                GetBucketVersioningRequest request = new GetBucketVersioningRequest
                {
                    BucketName = bucketName
                };

                GetBucketVersioningResponse response =
client.GetBucketVersioning(request);
                return response.VersioningConfig.Status;
            }
    }
```

```
}
```

Java

Untuk petunjuk tentang cara membuat dan menguji sampel kerja, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import java.io.IOException;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Region;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3Client;
import com.amazonaws.services.s3.model.AmazonS3Exception;
import com.amazonaws.services.s3.model.BucketVersioningConfiguration;
import com.amazonaws.services.s3.model.SetBucketVersioningConfigurationRequest;

public class BucketVersioningConfigurationExample {
    public static String bucketName = "*** bucket name ***";
    public static AmazonS3Client s3Client;

    public static void main(String[] args) throws IOException {
        s3Client = new AmazonS3Client(new ProfileCredentialsProvider());
        s3Client.setRegion(Region.getRegion(Regions.US_EAST_1));
        try {

            // 1. Enable versioning on the bucket.
         BucketVersioningConfiguration configuration =
           new BucketVersioningConfiguration().withStatus("Enabled");

   SetBucketVersioningConfigurationRequest setBucketVersioningConfigurationRequest
 =
     new SetBucketVersioningConfigurationRequest(bucketName,configuration);


 s3Client.setBucketVersioningConfiguration(setBucketVersioningConfigurationRequest);

   // 2. Get bucket versioning configuration information.
   BucketVersioningConfiguration conf =
 s3Client.getBucketVersioningConfiguration(bucketName);
    System.out.println("bucket versioning configuration status:    " +
 conf.getStatus());
```

```
        } catch (AmazonS3Exception amazonS3Exception) {
            System.out.format("An Amazon S3 error occurred. Exception: %s",
 amazonS3Exception.toString());
        } catch (Exception ex) {
            System.out.format("Exception: %s", ex.toString());
        }
    }
}
```

Python

Contoh kode Python berikut membuat bucket Amazon S3, mengaktifkannya untuk Penentuan Versi, dan mengonfigurasi siklus hidup yang membuat versi objek nonterkini kedaluwarsa setelah 7 hari.

```
def create_versioned_bucket(bucket_name, prefix):
    """
    Creates an Amazon S3 bucket, enables it for versioning, and configures a
 lifecycle
    that expires noncurrent object versions after 7 days.

    Adding a lifecycle configuration to a versioned bucket is a best practice.
    It helps prevent objects in the bucket from accumulating a large number of
    noncurrent versions, which can slow down request performance.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket_name: The name of the bucket to create.
    :param prefix: Identifies which objects are automatically expired under the
                   configured lifecycle rules.
    :return: The newly created bucket.
    """
    try:
        bucket = s3.create_bucket(
            Bucket=bucket_name,
            CreateBucketConfiguration={
                "LocationConstraint": s3.meta.client.meta.region_name
            },
        )
        logger.info("Created bucket %s.", bucket.name)
    except ClientError as error:
        if error.response["Error"]["Code"] == "BucketAlreadyOwnedByYou":
```

```
            logger.warning("Bucket %s already exists! Using it.", bucket_name)
            bucket = s3.Bucket(bucket_name)
        else:
            logger.exception("Couldn't create bucket %s.", bucket_name)
            raise

    try:
        bucket.Versioning().enable()
        logger.info("Enabled versioning on bucket %s.", bucket.name)
    except ClientError:
        logger.exception("Couldn't enable versioning on bucket %s.", bucket.name)
        raise

    try:
        expiration = 7
        bucket.LifecycleConfiguration().put(
            LifecycleConfiguration={
                "Rules": [
                    {
                        "Status": "Enabled",
                        "Prefix": prefix,
                        "NoncurrentVersionExpiration": {"NoncurrentDays":
expiration},
                    }
                ]
            }
        )
        logger.info(
            "Configured lifecycle to expire noncurrent versions after %s days "
            "on bucket %s.",
            expiration,
            bucket.name,
        )
    except ClientError as error:
        logger.warning(
            "Couldn't configure lifecycle on bucket %s because %s. "
            "Continuing anyway.",
            bucket.name,
            error,
        )

    return bucket
```

Mengonfigurasi penghapusan MFA

Saat bekerja dengan Penentuan Versi S3 di bucket Amazon S3, Anda dapat secara opsional menambahkan lapisan keamanan lainnya dengan mengonfigurasi bucket untuk mengaktifkan Penghapusan MFA (autentikasi multi-faktor). Saat melakukannya, pemilik bucket harus menyertakan dua bentuk autentikasi dalam setiap permintaan untuk menghapus sebuah versi atau mengubah status Penentuan Versi bucket.

Penghapusan MFA memerlukan autentikasi tambahan untuk salah satu dari operasi berikut:

- Mengubah status Penentuan Versi bucket Anda
- Menghapus versi objek secara permanen

Penghapusan MFA memerlukan dua bentuk autentikasi secara bersamaan:

- Kredensial keamanan Anda
- Gabungan nomor seri yang valid, spasi, dan kode enam digit yang ditampilkan di perangkat autentikasi yang disetujui

Penghapusan MFA memberikan keamanan tambahan jika, misalnya, kredensial keamanan Anda disusupi. Penghapusan MFA dapat membantu mencegah penghapusan bucket yang tidak disengaja dengan mengharuskan pengguna yang memulai tindakan penghapusan untuk membuktikan kepemilikan fisik perangkat MFA dengan kode MFA dan menambahkan lapisan gesek dan keamanan ekstra ke tindakan penghapusan.

Untuk mengidentifikasi bucket dengan penghapusan MFA yang aktif, Anda dapat menggunakan metrik Lensa Penyimpanan Amazon S3. Lensa Penyimpanan S3 adalah fitur analisis penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan objek. Untuk informasi selengkapnya, lihat Menilai aktivitas penyimpanan dan penggunaan Anda dengan Lensa Penyimpanan S3. Untuk daftar lengkap metrik, lihat Glosarium metrik Lensa Penyimpanan S3.

Pemilik bucket, Akun AWS yang membuat bucket (akun root), dan semua pengguna yang berwenang dapat mengaktifkan pembuatan versi. Namun, hanya pemilik bucket (akun root) yang dapat

mengaktifkan penghapusan MFA. Untuk informasi selengkapnya, lihat Mengamankan Akses AWS Menggunakan MFA di Blog Keamanan AWS .

> **Note**
>
> Untuk menggunakan MFA dengan Penentuan Versi, Anda mengaktifkan `MFA Delete`. Namun, Anda tidak dapat mengaktifkan `MFA Delete` menggunakan AWS Management Console. Anda harus menggunakan AWS Command Line Interface (AWS CLI) atau API. Untuk contoh menggunakan penghapusan MFA dengan Penentuan Versi, lihat bagian contoh dalam topik Mengaktifkan Penentuan Versi pada bucket.
>
> Anda tidak dapat menggunakan penghapusan MFA dengan konfigurasi siklus hidup. Untuk informasi selengkapnya tentang konfigurasi siklus hidup dan caranya berinteraksi dengan konfigurasi lain, lihat Siklus hidup dan konfigurasi bucket lainnya.

Untuk mengaktifkan atau menonaktifkan penghapusan MFA, Anda menggunakan API yang sama yang Anda gunakan untuk mengonfigurasi Penentuan Versi pada bucket. Amazon S3 menyimpan konfigurasi penghapusan MFA dalam subsumber daya Penentuan Versi yang menyimpan status Penentuan Versi bucket.

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Status>VersioningState</Status>
  <MfaDelete>MfaDeleteState</MfaDelete>
</VersioningConfiguration>
```

Untuk menggunakan penghapusan MFA, Anda dapat menggunakan perangkat keras atau perangkat MFA virtual untuk menghasilkan kode autentikasi. Contoh berikut menunjukkan kode autentikasi yang dihasilkan yang ditampilkan pada perangkat keras.

Penghapusan MFA dan akses API yang dilindungi MFA adalah fitur yang dimaksudkan untuk memberikan perlindungan untuk skenario yang berbeda. Anda mengonfigurasi penghapusan MFA di bucket untuk membantu memastikan bahwa data di bucket Anda tidak dapat terhapus secara tidak sengaja. Akses API yang dilindungi MFA digunakan untuk memberlakukan faktor autentikasi lain (kode MFA) ketika mengakses sumber daya Amazon S3 yang sensitif. Anda dapat meminta operasi

apa pun terhadap sumber daya Amazon S3 ini dilakukan dengan kredensial sementara yang dibuat menggunakan MFA. Sebagai contoh, lihat Membutuhkan MFA.

Untuk informasi selengkapnya tentang cara membeli dan mengaktifkan perangkat autentikasi, lihat Autentikasi multi-faktor.

Untuk mengaktifkan Penentuan Versi S3 dan mengkonfigurasi penghapusan MFA

Menggunakan AWS CLI

Contoh berikut mengaktifkan penghapusan Penentuan Versi S3 dan autentikasi multi-faktor (MFA) pada bucket.

```
aws s3api put-bucket-versioning --bucket example-s3-bucket1 --versioning-configuration
 Status=Enabled,MFADelete=Enabled --mfa "SERIAL 123456"
```

Penggunaan API REST

Untuk informasi selengkapnya tentang menentukan penghapusan MFA menggunakan Amazon S3 REST API, lihat Referensi API Layanan Penyimpanan Sederhana PutBucketVersioningAmazon.

Bekerja dengan objek di dalam bucket dengan dukungan Penentuan Versi

Objek yang disimpan di bucket Amazon S3 sebelum Anda mengatur status Penentuan Versi memiliki ID versi `null`. Saat Anda mengaktifkan Penentuan Versi, objek yang ada di bucket tidak berubah. Perubahannya adalah cara Amazon S3 menangani objek di masa mendatang.

Peralihan versi objek

Anda dapat menentukan aturan konfigurasi siklus hidup untuk objek yang memiliki siklus hidup yang ditetapkan dengan baik untuk transisi versi objek ke kelas penyimpanan S3 Glacier Flexible Retrieval pada waktu tertentu dalam masa aktif objek. Untuk informasi selengkapnya, lihat Mengelola siklus hidup penyimpanan Anda.

Topik dalam bagian ini menjelaskan berbagai operasi objek dalam bucket dengan dukungan Penentuan Versi. Untuk informasi selengkapnya tentang penentuan versi, lihat Menggunakan Penentuan Versi dalam bucket S3.

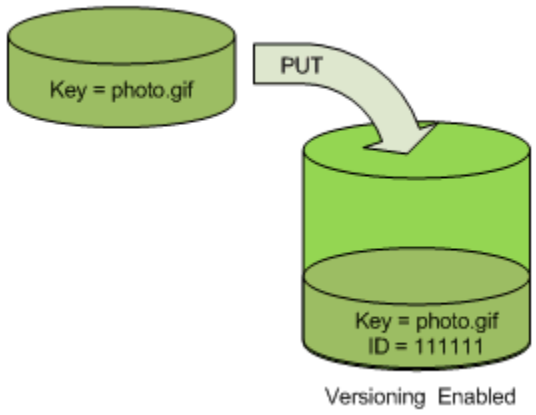
Topik

- Menambahkan objek ke bucket dengan dukungan Penentuan Versi
- Mencantumkan objek dalam bucket dengan dukungan Penentuan Versi
- Mengambil versi objek dari bucket dengan dukungan Penentuan Versi
- Menghapus versi objek dari bucket dengan dukungan Penentuan Versi
- Mengonfigurasi izin objek berversi

Menambahkan objek ke bucket dengan dukungan Penentuan Versi

Setelah Anda mengaktifkan Penentuan Versi di bucket, Amazon S3 secara otomatis menambahkan ID versi unik ke setiap objek yang disimpan (menggunakan `PUT`, `POST`, atau `CopyObject`) di bucket.

Gambar berikut menunjukkan bahwa Amazon S3 menambahkan ID versi unik ke objek saat ditambahkan ke bucket dengan dukungan Penentuan Versi.



> **Note**
>
> Nilai ID versi yang ditetapkan Amazon S3 adalah URL aman (dapat disertakan sebagai bagian dari URI).

Untuk informasi selengkapnya tentang penentuan versi, lihat Menggunakan Penentuan Versi dalam bucket S3. Anda dapat menambahkan versi objek ke bucket berkemampuan versi menggunakan konsol, AWS SDK, dan REST API.

Menggunakan konsol

Untuk petunjuk, lihat Mengunggah Objek.

Menggunakan AWS SDK

Untuk contoh mengunggah objek menggunakan AWS SDK untuk Java, .NET, dan PHP, lihat. Mengunggah Objek Contoh untuk mengunggah objek dalam bucket tanpa Penentuan Versi dan dengan Penentuan Versi yang diaktifkan adalah sama, meskipun dalam kasus bucket dengan dukungan Penentuan Versi, Amazon S3 menetapkan nomor versi. Jika tidak, nomor versi adalah null.

Untuk informasi tentang menggunakan AWS SDK lain, lihat Pusat AWS Pengembang.

Penggunaan API REST

Untuk menambahkan objek ke bucket dengan dukungan Penentuan Versi

1. Aktifkan Penentuan Versi di bucket menggunakan permintaan `PutBucketVersioning`.

   Untuk informasi selengkapnya, lihat PutBucketVersioning dalam Referensi API Amazon Simple Storage Service.
2. Kirim permintaan `PUT`, `POST`, atau `CopyObject` untuk menyimpan objek dalam bucket.

Saat Anda menambahkan objek ke bucket dengan dukungan Penentuan Versi, Amazon S3 mengembalikan ID versi objek di header respons `x-amz-version-id`, seperti yang ditunjukkan dalam contoh berikut.

```
x-amz-version-id: 3/L4kqtJlcpXroDTDmJ+rmSpXd3dIbrHY
```

Mencantumkan objek dalam bucket dengan dukungan Penentuan Versi

Bagian ini memberikan contoh daftar versi objek dari bucket dengan dukungan Penentuan Versi. Amazon S3 menyimpan informasi versi objek dalam subsumber daya versi yang dikaitkan dengan bucket. Untuk informasi selengkapnya, lihat Opsi konfigurasi bucket. Untuk mencantumkan objek dalam bucket dengan dukungan Penentuan Versi, Anda memerlukan izin `ListBucketVersions`.

Menggunakan konsol S3

Ikuti langkah-langkah ini untuk menggunakan konsol Amazon S3 guna melihat berbagai versi objek.

Untuk melihat beberapa versi objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di dalam daftar Bucket, pilih nama bucket yang berisi objek.
3. Untuk melihat daftar versi objek di bucket, pilih tombol Tampilkan versi.

   Untuk setiap versi objek, konsol menunjukkan ID versi unik, tanggal dan waktu versi objek dibuat, dan properti lainnya. (Objek yang disimpan di bucket sebelum Anda mengatur status Penentuan Versi memiliki ID versi null.)

   Untuk mencantumkan objek tanpa versi, pilih tombol Buat daftar versi.

Anda juga dapat melihat, mengunduh, dan menghapus versi objek di panel ikhtisar objek di konsol. Untuk informasi selengkapnya, lihat Melihat gambaran umum objek di konsol Amazon S3.

> **Note**
>
> Untuk mengakses versi objek yang lebih lama dari 300 versi, Anda harus menggunakan AWS CLI atau URL objek.

> **Important**
>
> Anda dapat membatalkan penghapusan objek hanya jika objek tersebut dihapus sebagai versi terbaru (saat ini). Anda tidak dapat membatalkan penghapusan objek versi sebelumnya. Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.

Menggunakan AWS SDK

Contoh dalam bagian ini menunjukkan cara mengambil daftar objek dari bucket dengan dukungan Penentuan Versi. Setiap permintaan akan menampilkan hingga 1.000 versi, kecuali jika Anda menentukan nomor yang lebih rendah. Jika bucket berisi lebih banyak versi dari batas ini, Anda mengirimkan serangkaian permintaan untuk mengambil daftar semua versi. Proses mengembalikan hasil dalam "halaman” disebut penomoran halaman.

Untuk menunjukkan cara kerja penomoran halaman, contoh membatasi masing-masing respons pada dua versi objek. Setelah mengambil halaman pertama dari hasil, masing-masing contoh memeriksa untuk menentukan apakah daftar versi terpotong. Jika iya, contoh ini terus mengambil halaman hingga semua versi diambil.

> **Note**
>
> Contoh berikut juga bekerja dengan bucket tanpa Penentuan Versi aktif, atau untuk objek yang tidak memiliki versi tersendiri. Dalam kasus tersebut, Amazon S3 mengembalikan daftar objek dengan ID versi `null`.

Untuk informasi tentang menggunakan AWS SDK lain, lihat Pusat AWS Pengembang.

Java

Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListVersionsRequest;
import com.amazonaws.services.s3.model.S3VersionSummary;
import com.amazonaws.services.s3.model.VersionListing;

public class ListKeysVersioningEnabledBucket {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Retrieve the list of versions. If the bucket contains more versions
            // than the specified maximum number of results, Amazon S3 returns
            // one page of results per request.
            ListVersionsRequest request = new ListVersionsRequest()
                    .withBucketName(bucketName)
```

```
                    .withMaxResults(2);
            VersionListing versionListing = s3Client.listVersions(request);
            int numVersions = 0, numPages = 0;
            while (true) {
                numPages++;
                for (S3VersionSummary objectSummary :
 versionListing.getVersionSummaries()) {
                    System.out.printf("Retrieved object %s, version %s\n",
                            objectSummary.getKey(),
                            objectSummary.getVersionId());
                    numVersions++;
                }
                // Check whether there are more pages of versions to retrieve. If
                // there are, retrieve them. Otherwise, exit the loop.
                if (versionListing.isTruncated()) {
                    versionListing =
 s3Client.listNextBatchOfVersions(versionListing);
                } else {
                    break;
                }
            }
            System.out.println(numVersions + " object versions retrieved in " +
 numPages + " pages");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
```

```
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class ListObjectsVersioningEnabledBucketTest
    {
        static string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main(string[] args)
        {
            s3Client = new AmazonS3Client(bucketRegion);
            GetObjectListWithAllVersionsAsync().Wait();
        }

        static async Task GetObjectListWithAllVersionsAsync()
        {
            try
            {
                ListVersionsRequest request = new ListVersionsRequest()
                {
                    BucketName = bucketName,
                    // You can optionally specify key name prefix in the request
                    // if you want list of object versions of a specific object.

                    // For this example we limit response to return list of 2
 versions.
                    MaxKeys = 2
                };
                do
                {
                    ListVersionsResponse response = await
 s3Client.ListVersionsAsync(request);
                    // Process response.
                    foreach (S3ObjectVersion entry in response.Versions)
                    {
                        Console.WriteLine("key = {0} size = {1}",
                            entry.Key, entry.Size);
                    }
```

```
                    // If response is truncated, set the marker to get the next
                    // set of keys.
                    if (response.IsTruncated)
                    {
                        request.KeyMarker = response.NextKeyMarker;
                        request.VersionIdMarker = response.NextVersionIdMarker;
                    }
                    else
                    {
                        request = null;
                    }
                } while (request != null);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

Penggunaan API REST

Example — Mencantumkan semua versi objek dalam bucket

Untuk mencantumkan semua versi dari objek dalam bucket, Anda menggunakan subsumber daya `versions` di permintaan `GET Bucket`. Amazon S3 dapat mengambil maksimum 1.000 objek, dan setiap versi objek dihitung sepenuhnya sebagai objek. Oleh karena itu, jika bucket berisi dua kunci (misalnya, `photo.gif` dan `picture.jpg`), dan kunci pertama memiliki 990 versi dan kunci kedua memiliki 400 versi, satu permintaan akan mengambil seluruh 990 versi `photo.gif` dan hanya 10 versi terbaru dari `picture.jpg`.

Amazon S3 mengembalikan versi objek dalam urutan penyimpanannya, dengan versi yang disimpan paling akhir dikembalikan terlebih dahulu.

Di permintaan `GET Bucket`, sertakan subsumber daya `versions`.

```
GET /?versions HTTP/1.1
Host: bucketName.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 +0000
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Example — Mengambil semua versi kunci

Untuk mengambil subset versi objek, Anda menggunakan parameter permintaan untuk `GET Bucket`. Untuk informasi selengkapnya, lihat GET Bucket.

1. Atur parameter `prefix` ke kunci objek yang ingin Anda ambil.
2. Kirim permintaan `GET Bucket` menggunakan subsumber daya `versions` dan `prefix`.

   ```
   GET /?versions&prefix=objectName HTTP/1.1
   ```

Example — Mengambil objek menggunakan prefiks

Contoh berikut mengambil objek yang kuncinya adalah atau dimulai dengan `myObject`.

```
GET /?versions&prefix=myObject HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Anda dapat menggunakan parameter permintaan lain untuk mengambil subset semua versi objek. Untuk informasi selengkapnya, lihat GET Bucket dalam Referensi API Amazon Simple Storage Service.

Example — Mengambil daftar objek tambahan jika respons terpotong

Jika jumlah objek yang dapat dikembalikan dalam permintaan `GET` melebihi nilai `max-keys`, respons mengandung `<isTruncated>true</isTruncated>`, dan menyertakan kunci pertama (dalam `NextKeyMarker`) dan ID versi pertama (dalam `NextVersionIdMarker`) yang memenuhi permintaan, tetapi tidak dikembalikan. Anda menggunakan nilai yang dikembalikan sebagai posisi awal pada permintaan berikutnya untuk mengambil objek tambahan yang memenuhi permintaan `GET`.

Gunakan proses berikut untuk mengambil objek tambahan yang sesuai dengan permintaan `GET Bucket versions` dari bucket. Untuk informasi selengkapnya tentang `key-marker`, `version-id-marker`, `NextKeyMarker`, dan `NextVersionIdMarker`, lihat GET Bucket di Referensi API Amazon Simple Storage Service.

Berikut ini adalah tanggapan tambahan yang memenuhi permintaan `GET` asli:

- Tetapkan nilai dari `key-marker` ke kunci yang dikembalikan di `NextKeyMarker` dalam respons sebelumnya.
- Tetapkan nilai dari `version-id-marker` ke ID versi yang dikembalikan di `NextVersionIdMarker` dalam respons sebelumnya.
- Kirim permintaan `GET Bucket versions` menggunakan `key-marker` dan `version-id-marker`.

Example — Mengambil objek yang dimulai dengan kunci dan ID versi tertentu

```
GET /?versions&key-marker=myObject&version-id-marker=298459348571 HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Menggunakan AWS CLI

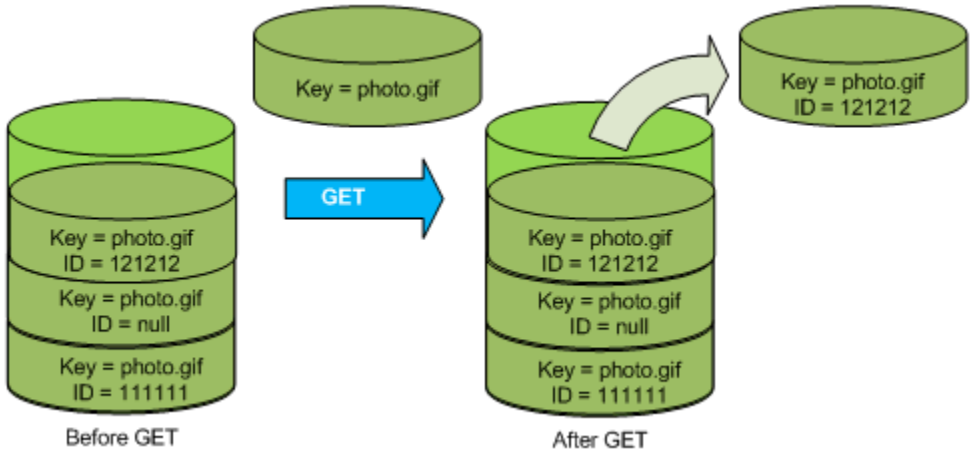
Perintah berikut mengembalikan metadata tentang semua versi objek dalam bucket.

```
aws s3api list-object-versions --bucket example-s3-bucket1
```

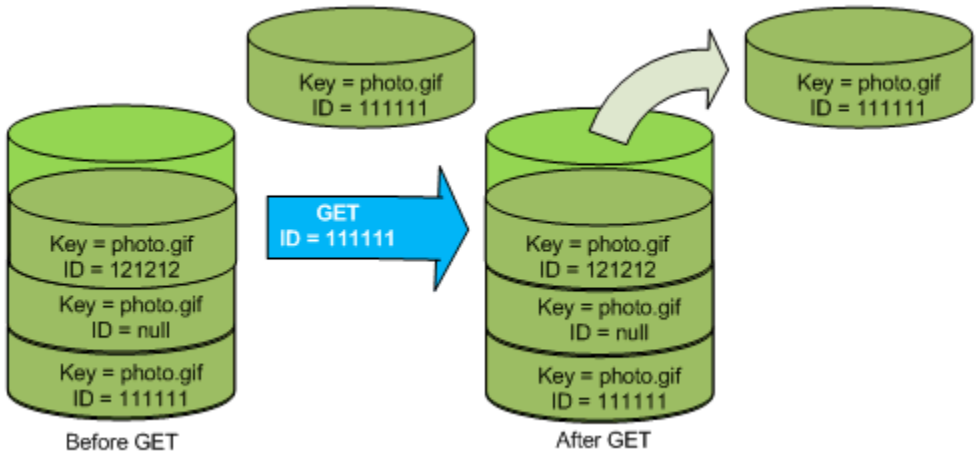
Untuk informasi selengkapnya tentang `list-object-versions`, lihat list-object-versions di Referensi Perintah AWS CLI .

Mengambil versi objek dari bucket dengan dukungan Penentuan Versi

Penentuan Versi di Amazon S3 adalah cara menyimpan beberapa varian objek dalam bucket yang sama. Permintaan `GET` sederhana mengambil versi saat ini dari sebuah objek. Gambar berikut menunjukkan bagaimana `GET` mengembalikan versi objek saat ini, `photo.gif`.



Untuk mengambil versi tertentu, Anda harus menentukan ID versinya. Gambar berikut menunjukkan bahwa permintaan `GET versionId` mengambil versi objek yang ditentukan (tidak harus versi saat ini).



Anda dapat mengambil versi objek di Amazon S3 menggunakan konsol AWS , SDK, atau REST API.

> **Note**
>
> Untuk mengakses versi objek yang lebih lama dari 300 versi, Anda harus menggunakan AWS CLI atau URL objek.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di dalam daftar Bucket, pilih nama bucket yang berisi objek.
3. Di daftar Objek, pilih nama objek.
4. Pilih Versi.

   Amazon S3 menampilkan semua versi untuk objek tersebut.
5. Pilih kotak centang di sebelah ID Versi untuk versi yang ingin Anda ambil.
6. Pilih Tindakan, pilih Unduh, dan simpan objek.

Anda juga dapat melihat, mengunduh, dan menghapus versi objek di panel ikhtisar objek. Untuk informasi selengkapnya, lihat Melihat gambaran umum objek di konsol Amazon S3.

> **Important**
>
> Anda dapat membatalkan penghapusan objek hanya jika objek tersebut dihapus sebagai versi terbaru (saat ini). Anda tidak dapat membatalkan penghapusan objek versi sebelumnya. Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.

Menggunakan AWS SDK

Contoh untuk pengunggahan objek dalam bucket tanpa versi dan dengan dukungan Penentuan Versi adalah sama. Namun, untuk bucket dengan dukungan Penentuan Versi, Amazon S3 menetapkan nomor versi. Jika tidak, nomor versi adalah null.

Untuk contoh mengunduh objek menggunakan AWS SDK untuk Java, .NET, dan PHP, lihat Mengunduh objek.

Untuk contoh mencantumkan versi objek yang menggunakan AWS SDK untuk.NET dan Rust, lihat Daftar versi objek di bucket Amazon S3.

Penggunaan API REST

Untuk mengambil versi objek tertentu

1. Atur `versionId` ke ID versi objek yang ingin Anda ambil.
2. Kirim permintaan `GET Object versionId`.

Example — Mengambil objek berversi

Permintaan berikut mengambil versi `L4kqtJlcpXroDTDmpUMLUo` dari `my-image.jpg`.

```
GET /my-image.jpg?versionId=L4kqtJlcpXroDTDmpUMLUo HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Anda hanya dapat mengambil metadata dari suatu objek (bukan konten). Untuk informasi, lihat the section called “Mengambil metadata versi”.

Untuk informasi tentang memulihkan versi objek sebelumnya, lihat the section called “Memulihkan versi sebelumnya”.

Mengambil metadata versi objek

Jika Anda hanya ingin mengambil metadata suatu objek (dan bukan kontennya), Anda menggunakan operasi `HEAD`. Secara default, Anda mendapatkan metadata versi terbaru. Untuk mengambil metadata versi objek tertentu, Anda menentukan ID versinya.

Untuk mengambil metadata versi objek

1. Atur `versionId` ke ID versi objek yang ingin Anda ambil metadatanya.
2. Kirim permintaan `HEAD Object versionId`.

Example — Mengambil metadata dari objek berversi

Permintaan berikut mengambil metadata versi `3HL4kqCxf3vjVBH40Nrjfkd` dari `my-image.jpg`.

```
HEAD /my-image.jpg?versionId=3HL4kqCxf3vjVBH40Nrjfkd HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Berikut ini menunjukkan respons sampel.

```
HTTP/1.1 200 OK
x-amz-id-2: ef8yU9AS1ed4OpIszj7UDNEHGran
x-amz-request-id: 318BC8BC143432E5
x-amz-version-id: 3HL4kqtJlcpXroDTDmjVBH40Nrjfkd
```

```
Date: Wed, 28 Oct 2009 22:32:00 GMT
Last-Modified: Sun, 1 Jan 2006 12:00:00 GMT
ETag: "fba9dede5f27731c9771645a39863328"
Content-Length: 434234
Content-Type: text/plain
Connection: close
Server: AmazonS3
```

Memulihkan versi sebelumnya

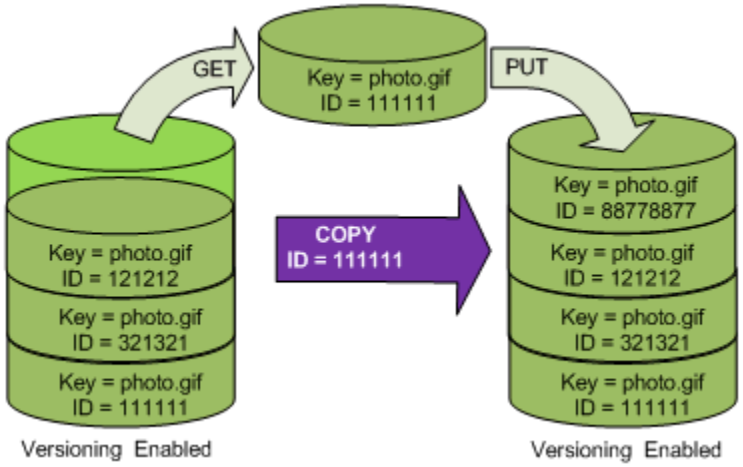
Anda dapat menggunakan Penentuan Versi untuk mengambil versi sebelumnya dari objek. Ada dua pendekatan untuk melakukannya:

- Salin versi objek sebelumnya ke dalam bucket yang sama.

  Objek yang disalin menjadi versi saat ini dari objek tersebut dan semua versi objek dipertahankan.
- Hapus versi objek saat ini secara permanen.

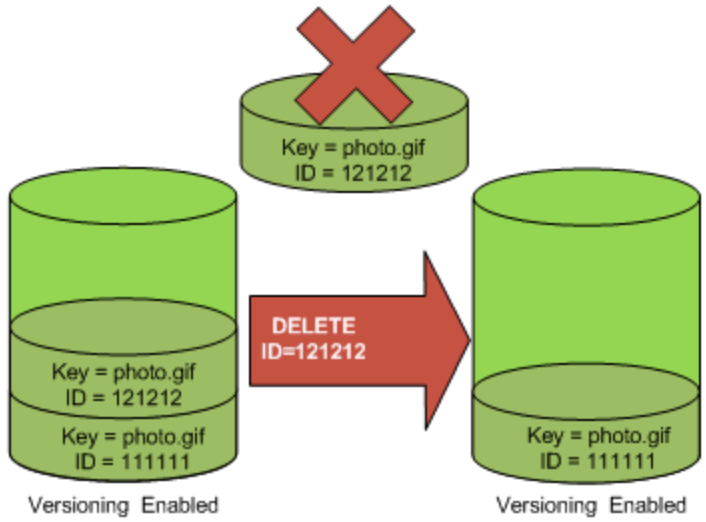
  Ketika versi objek saat ini dihapus, Anda, akibatnya, mengubah versi sebelumnya menjadi versi saat ini dari objek tersebut.

Karena semua versi objek dipertahankan, Anda dapat membuat versi yang lebih awal dari versi saat ini dengan menyalin versi tertentu dari objek tersebut ke dalam bucket yang sama. Pada gambar berikut, objek sumber (ID = 111111) disalin ke dalam bucket yang sama. Amazon S3 memasok ID baru (88778877) dan menjadi versi objek saat ini. Jadi, bucket memiliki objek versi asli (111111) dan salinannya (88778877). Untuk informasi selengkapnya tentang mendapatkan versi sebelumnya dan kemudian mengunggahnya untuk menjadikannya versi saat ini, lihat Mengambil versi objek dari bucket dengan dukungan Penentuan Versi dan Mengunggah objek.



GET berikutnya mengambil versi 88778877.

Gambar berikut menunjukkan bagaimana menghapus versi saat ini (121212) dari suatu objek menjadikan versi sebelumnya (111111) sebagai objek saat ini. Untuk informasi selengkapnya tentang menghapus objek, lihat Menghapus satu objek.



GET berikutnya mengambil versi 111111.

> **Note**
>
> Untuk mengembalikan versi objek dalam batch, Anda dapat menggunakan operasi `CopyObject`. Operasi salin dapat menyalin setiap objek yang ditentukan dalam manifes. Namun, ketahuilah bahwa objek tidak selalu disalin dalam urutan yang sama seperti yang tercantum dalam manifes. Untuk bucket berversi, jika mempertahankan urutan versi terkini/nonterkini bersifat penting, Anda harus menyalin semua versi nonterkini terlebih dahulu. Kemudian, setelah pekerjaan pertama selesai, salin versi terbaru pada pekerjaan berikutnya.

Untuk mengembalikan versi objek sebelumnya

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Bucket, pilih nama bucket yang berisi objek.
3. Di daftar Objek, pilih nama objek.

4. Pilih Versi.

   Amazon S3 menampilkan semua versi untuk objek tersebut.
5. Pilih kotak centang di sebelah ID Versi untuk versi yang ingin Anda ambil.
6. Pilih Tindakan, pilih Unduh, dan simpan objek.

Anda juga dapat melihat, mengunduh, dan menghapus versi objek di panel ikhtisar objek. Untuk informasi selengkapnya, lihat Melihat gambaran umum objek di konsol Amazon S3.

> **Important**
>
> Anda dapat membatalkan penghapusan objek hanya jika objek tersebut dihapus sebagai versi terbaru (saat ini). Anda tidak dapat membatalkan penghapusan objek versi sebelumnya. Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.

Menggunakan AWS SDK

Untuk informasi tentang menggunakan AWS SDK lain, lihat Pusat AWS Pengembang.

Python

Contoh kode Python berikut memulihkan versi objek berversi sebelumnya dengan menghapus semua versi yang terjadi setelah versi rollback yang ditentukan.

```
def rollback_object(bucket, object_key, version_id):
    """
    Rolls back an object to an earlier version by deleting all versions that
    occurred after the specified rollback version.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that holds the object to roll back.
    :param object_key: The object to roll back.
    :param version_id: The version ID to roll back to.
    """
    # Versions must be sorted by last_modified date because delete markers are
    # at the end of the list even when they are interspersed in time.
    versions = sorted(
```

```
        bucket.object_versions.filter(Prefix=object_key),
        key=attrgetter("last_modified"),
        reverse=True,
    )

    logger.debug(
        "Got versions:\n%s",
        "\n".join(
            [
                f"\t{version.version_id}, last modified {version.last_modified}"
                for version in versions
            ]
        ),
    )

    if version_id in [ver.version_id for ver in versions]:
        print(f"Rolling back to version {version_id}")
        for version in versions:
            if version.version_id != version_id:
                version.delete()
                print(f"Deleted version {version.version_id}")
            else:
                break

        print(f"Active version is now {bucket.Object(object_key).version_id}")
    else:
        raise KeyError(
            f"{version_id} was not found in the list of versions for "
f"{object_key}."
        )
```

Menghapus versi objek dari bucket dengan dukungan Penentuan Versi

Anda dapat menghapus versi objek dari bucket Amazon S3 kapan pun Anda inginkan. Selain itu, Anda juga dapat menentukan aturan konfigurasi siklus hidup untuk objek yang memiliki siklus hidup yang ditentukan dengan baik guna meminta Amazon S3 untuk menghentikan versi objek saat ini atau menghapus versi objek nonterkini secara permanen. Ketika bucket Anda dalam status Penentuan Versi aktif atau Penentuan Versi ditangguhkan, tindakan konfigurasi siklus hidup bekerja sebagai berikut:

- Tindakan `Expiration` berlaku untuk versi objek saat ini. Alih-alih menghapus versi objek saat ini, Amazon S3 mempertahankan versi saat ini sebagai versi bukan saat ini dengan menambahkan penanda hapus, yang kemudian menjadi versi saat ini.
- Tindakan `NoncurrentVersionExpiration` tersebut berlaku untuk versi objek nonterkini, dan Amazon S3 secara permanen menghapus versi objek ini. Anda tidak dapat memulihkan kembali objek yang dihapus secara permanen.

Untuk informasi selengkapnya tentang Siklus Hidup S3, lihat Mengelola siklus hidup penyimpanan Anda dan Contoh konfigurasi Siklus Hidup S3.

Untuk melihat berapa banyak versi objek saat ini dan bukan saat ini yang dimiliki bucket, Anda dapat menggunakan metrik Lensa Penyimpanan Amazon S3. Lensa Penyimpanan S3 adalah fitur analisis penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan objek. Untuk informasi selengkapnya, lihat Menggunakan Lensa Penyimpanan S3 untuk mengoptimalkan biaya penyimpanan Anda. Untuk daftar lengkap metrik, lihat glosarium metrik Lensa Penyimpanan S3.

> **Note**
>
> Tarif Amazon S3 normal berlaku untuk setiap versi objek yang disimpan dan ditransfer, termasuk versi objek noncurrent. Untuk informasi selengkapnya, lihat Harga Amazon S3.

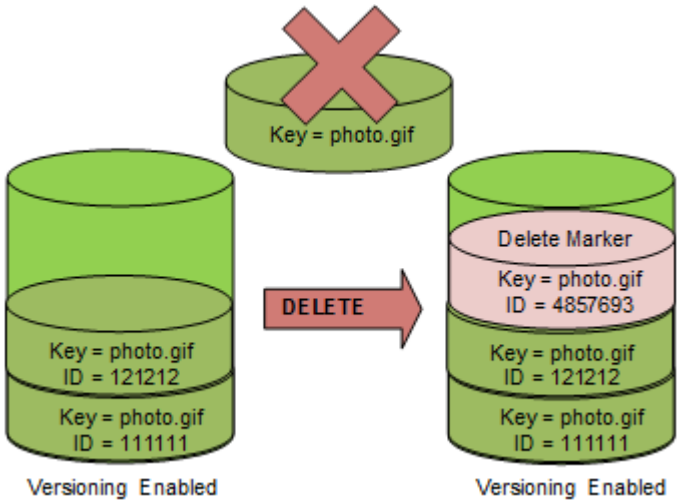
Menghapus kasus penggunaan permintaan

Permintaan `DELETE` memiliki kasus penggunaan berikut:

- Saat Penentuan Versi diaktifkan, `DELETE` sederhana tidak dapat menghapus objek secara permanen. (Permintaan `DELETE` sederhana adalah permintaan yang tidak menentukan ID versi.) Alih-alih demikian, Amazon S3 memasukkan penanda hapus dalam bucket, dan marker tersebut menjadi versi objek saat ini dengan ID baru.

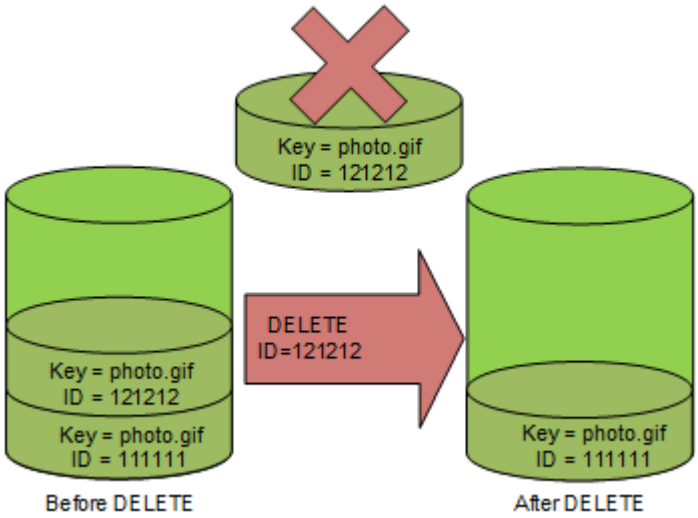
  Saat Anda mencoba untuk `GET` objek yang versinya saat ini adalah penanda hapus, Amazon S3 berperilaku seolah-olah objek telah dihapus (meskipun belum dihapus) dan mengembalikan pesan kesalahan 404. Untuk informasi selengkapnya, lihat Bekerja dengan penanda hapus.

  Gambar berikut menunjukkan bahwa `DELETE` sederhana tidak benar-benar menghapus objek tertentu. Sebaliknya, Amazon S3 menyisipkan penanda hapus.



- Untuk menghapus objek dengan Penentuan Versi secara permanen, Anda harus menggunakan `DELETE Object versionId`.

  Gambar berikut menunjukkan bahwa menghapus versi objek tertentu secara permanen menghapus objek tersebut.



Untuk menghapus versi objek

Anda dapat menghapus versi objek di Amazon S3 menggunakan konsol, AWS SDK, REST API, atau file. AWS Command Line Interface

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di dalam daftar Bucket, pilih nama bucket yang berisi objek.
3. Di daftar Objek, pilih nama objek.
4. Pilih Versi.

   Amazon S3 menampilkan semua versi untuk objek tersebut.
5. Pilih kotak centang di samping ID Versi untuk versi yang ingin Anda hapus secara permanen.
6. Pilih Hapus.
7. Di Hapus objek secara permanen?, masukkan **`permanently delete`**.

   > **Warning**
   >
   > Saat Anda menghapus versi objek secara permanen, tindakan itu tidak dapat dibatalkan.

8. Pilih Hapus objek.

   Amazon S3 menghapus versi objek.

Menggunakan AWS SDK

Untuk contoh menghapus objek menggunakan AWS SDK untuk Java, .NET, dan PHP, lihat. Menghapus objek Amazon S3 Contoh untuk menghapus objek dalam bucket tanpa versi dan dengan dukungan Penentuan Versi adalah sama. Namun, untuk bucket dengan dukungan Penentuan Versi, Amazon S3 menetapkan nomor versi. Jika tidak, nomor versi adalah null.

Untuk informasi tentang menggunakan AWS SDK lain, lihat Pusat AWS Pengembang.

Python

Contoh kode Python berikut secara permanen menghapus objek berversi dengan menghapus semua versinya.

```
def permanently_delete_object(bucket, object_key):
    """
    Permanently deletes a versioned object by deleting all of its versions.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that contains the object.
    :param object_key: The object to delete.
```

```
    """
    try:
        bucket.object_versions.filter(Prefix=object_key).delete()
        logger.info("Permanently deleted all versions of object %s.", object_key)
    except ClientError:
        logger.exception("Couldn't delete all versions of %s.", object_key)
        raise
```

Penggunaan API REST

Untuk menghapus versi objek tertentu

- Di `DELETE`, tentukan ID versi.

Example — Menghapus versi tertentu

Contoh berikut menghapus versi `UIORUnfnd89493jJFJ` dari `photo.gif`.

```
DELETE /photo.gif?versionId=UIORUnfnd89493jJFJ HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 12 Oct 2009 17:50:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:xQE0diMbLRepdf3YB+FIEXAMPLE=
Content-Type: text/plain
Content-Length: 0
```

Menggunakan AWS CLI

Perintah berikut menghapus objek bernama test.txt dari bucket bernama *`example-s3-bucket1`*. Untuk menghapus versi objek tertentu, Anda harus menjadi pemilik bucket dan harus menggunakan subsumber daya Id versi.

```
aws s3api delete-object --bucket example-s3-bucket1 --key test.txt --version-
id versionID
```

Untuk informasi selengkapnya tentang `delete-object`, lihat delete-object di Referensi Perintah AWS CLI .

Untuk informasi selengkapnya tentang menghapus versi objek, lihat topik berikut ini:

- Bekerja dengan penanda hapus
- Menghapus penanda hapus untuk membuat versi yang lebih lama menjadi versi saat ini
- Menghapus objek dari bucket dengan dukungan penghapusan MFA

Bekerja dengan penanda hapus

Penanda hapus di Amazon S3 adalah placeholder (atau penanda) untuk objek berversi yang ditentukan dalam permintaan `DELETE` sederhana. Permintaan `DELETE` sederhana adalah permintaan yang tidak menentukan ID versi. Karena objek berada dalam bucket dengan dukungan Penentuan Versi, objek tidak dihapus. Namun, penanda hapus membuat Amazon S3 berperilaku seolah-olah objek telah dihapus. Anda dapat menggunakan panggilan `DELETE` API Amazon S3 pada penanda hapus. Untuk melakukan ini, Anda harus membuat `DELETE` permintaan dengan menggunakan pengguna AWS Identity and Access Management (IAM) atau peran dengan izin yang sesuai.

Penanda hapus memiliki nama kunci (atau kunci) dan ID versi seperti objek lainnya. Namun, penanda hapus berbeda dengan objek lain dalam hal berikut:

- Penanda hapus tidak memiliki data yang terkait dengannya.
- Penanda penghapusan tidak terkait dengan nilai daftar kontrol akses (ACL).
- Jika Anda mengeluarkan permintaan `GET` untuk penanda hapus, permintaan `GET` tidak mengambil apa pun karena penanda hapus tidak memiliki data. Khususnya, ketika permintaan `GET` Anda tidak menentukan `versionId`, Anda mendapatkan pesan kesalahan 404 (Tidak Ditemukan).

Penanda hapus menambah sedikit biaya penyimpanan di Amazon S3. Ukuran penyimpanan penanda hapus sama dengan ukuran nama kunci dari penanda hapus. Nama kunci adalah deretan karakter Unicode. Pengodean UTF-8 untuk nama kunci menambahkan 1-4 byte penyimpanan ke bucket Anda untuk setiap karakter dalam nama. Penanda hapus disimpan di kelas penyimpanan S3 Standard.

Jika ingin mengetahui berapa banyak penanda hapus yang Anda miliki dan kelas penyimpanan apa yang disimpan, Anda dapat menggunakan Lensa Penyimpanan Amazon S3. Untuk informasi selengkapnya, lihat Menilai aktivitas penyimpanan dan penggunaan Anda dengan Amazon S3 Storage Lens dan Glosarium metrik Amazon S3 Storage Lens.

Untuk informasi selengkapnya tentang nama kunci, lihat Membuat nama kunci objek. Untuk informasi tentang menghapus penanda hapus, lihat Mengelola penanda hapus.

Hanya Amazon S3 yang dapat membuat penanda hapus, dan hal ini dilakukan setiap kali Anda mengirim permintaan `DeleteObject` atas objek dalam bucket dengan dukungan Penentuan Versi atau Penentuan Versi ditangguhkan. Objek yang ditentukan dalam permintaan `DELETE` tidak benar-benar dihapus. Alih-alih demikian, penanda hapus menjadi versi objek saat ini. Nama kunci objek (atau kunci) menjadi kunci dari penanda hapus.

Saat Anda mendapatkan objek tanpa menentukan `versionId` dalam permintaan, jika versinya saat ini adalah penanda hapus, Amazon S3 akan merespons dengan berikut:

- Pesan kesalahan 404 (Tidak Ditemukan)
- Header respons, `x-amz-delete-marker: true`

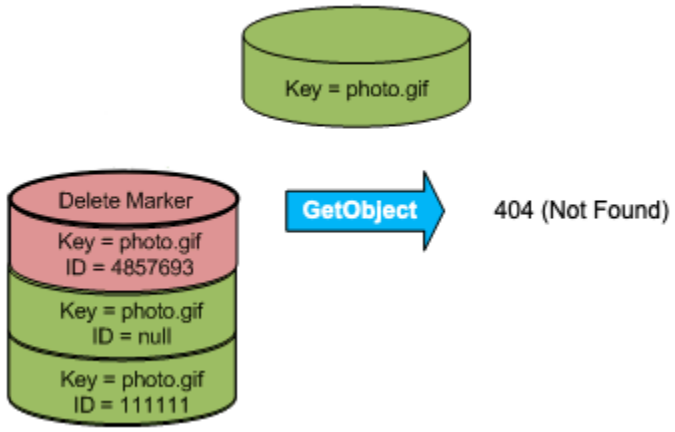
Saat Anda mendapatkan objek dengan menentukan `versionId` dalam permintaan, jika versi yang ditentukan adalah penanda hapus, Amazon S3 akan merespons dengan berikut:

- Pesan kesalahan 405 (Metode Tidak Diizinkan)
- Header respons, `x-amz-delete-marker: true`
- Header respons, `Last-Modified: timestamp` (hanya saat menggunakan operasi HeadObjectatau GetObjectAPI)

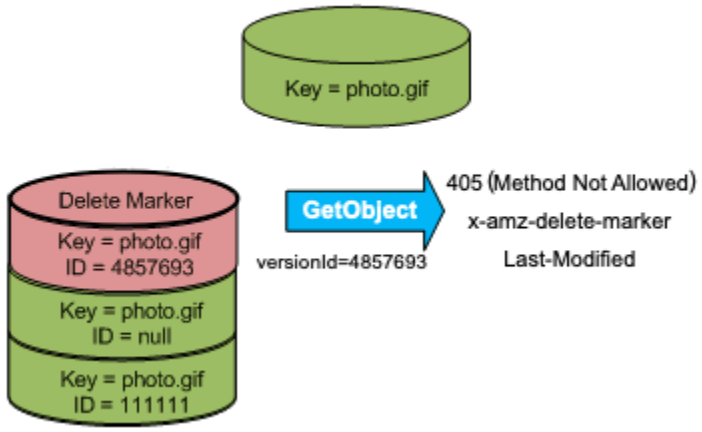
Header respons `x-amz-delete-marker: true` memberi tahu bahwa objek yang diakses adalah penanda hapus. Header respons ini tidak pernah mengembalikan `false`, karena ketika nilainya adalah `false`, versi objek saat ini atau yang ditentukan bukanlah penanda hapus.

Header respons `Last-Modified` menyediakan waktu pembuatan penanda hapus.

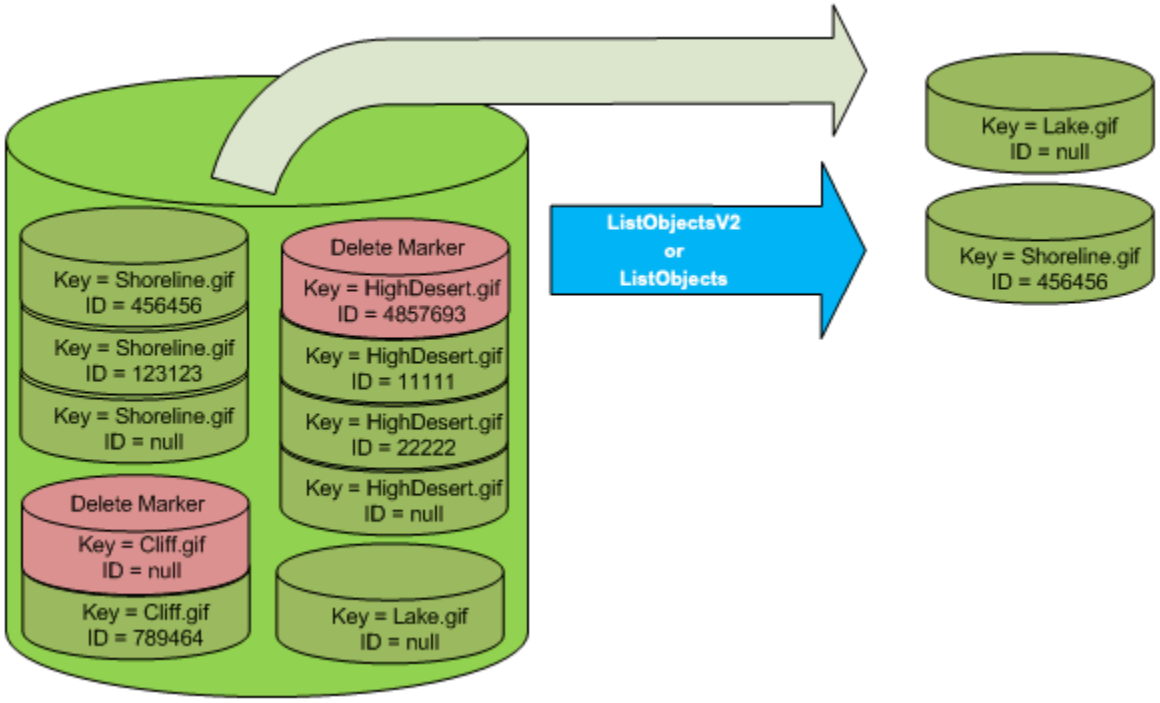
Gambar berikut menunjukkan cara panggilan API `GetObject` pada sebuah objek yang versinya saat ini adalah penanda hapus merespons dengan pesan kesalahan 404 (Tidak Ditemukan) dan header respons menyertakan `x-amz-delete-marker: true`.



Jika Anda melakukan panggilan `GetObject` pada objek dengan menentukan `versionId` dalam permintaan Anda, dan jika versi yang ditentukan adalah penanda hapus, Amazon S3 merespons dengan pesan kesalahan 405 (Metode Tidak Diizinkan) dan header respons menyertakan `x-amz-delete-marker: true` dan `Last-Modified: timestamp`.



Satu-satunya cara untuk mencantumkan penanda hapus (dan versi lain dari objek) adalah dengan menggunakan subsumber daya `versions` di permintaan ListObjectVersions. Gambar berikut menunjukkan bahwa permintaan ListObjectsV2 atau ListObjects tidak mengembalikan objek yang versinya saat ini adalah penanda hapus.



Mengelola penanda hapus

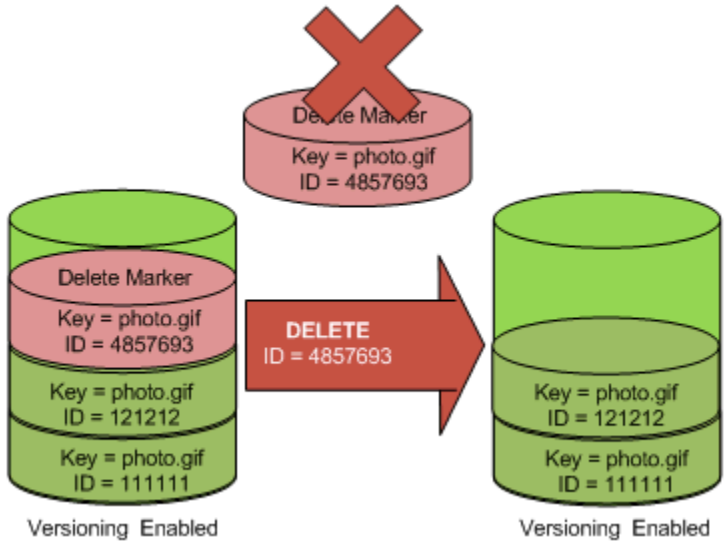
Mengonfigurasi siklus hidup untuk membersihkan penanda hapus kedaluwarsa secara otomatis

Penanda hapus objek yang kedaluwarsa adalah penanda yang semua versi objeknya dihapus dan hanya satu penanda hapus yang tersisa. Jika konfigurasi siklus hidup diatur untuk menghapus versi saat ini, atau tindakan `ExpiredObjectDeleteMarker` diatur secara eksplisit, Amazon S3 menghapus penanda hapus objek yang kedaluwarsa. Sebagai contoh, lihat Contoh 7: Menghapus penanda hapus objek kedaluwarsa.

Menghapus penanda hapus untuk membuat versi yang lebih lama menjadi versi saat ini

Saat Anda menghapus sebuah objek dalam bucket dengan dukungan Penentuan Versi, semua versi tetap berada dalam bucket, dan Amazon S3 membuat penanda hapus untuk objek tersebut. Untuk membatalkan penghapusan objek, Anda harus menghapus penanda hapus ini. Untuk informasi selengkapnya tentang Penentuan Versi dan penanda hapus, lihat Menggunakan Penentuan Versi dalam bucket S3.

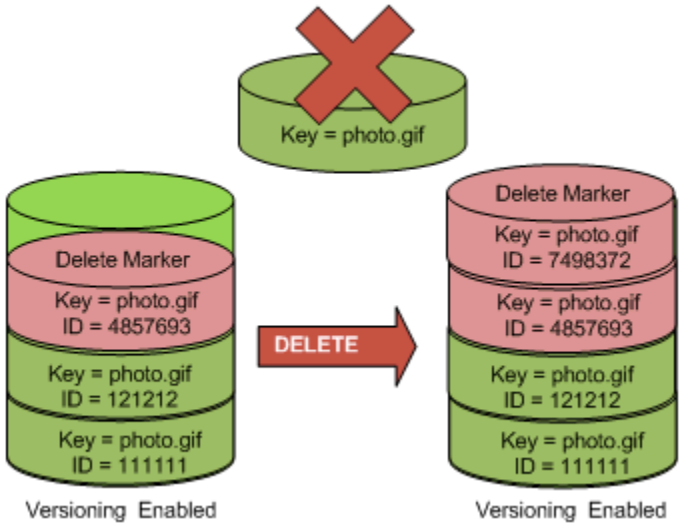
Untuk menghapus penanda hapus secara permanen, Anda harus menyertakan ID versinya dalam permintaan `DeleteObject versionId`. Gambar berikut menunjukkan cara permintaan `DeleteObject versionId` menghapus penanda hapus secara permanen.



Efek dari menghapus penanda hapus adalah bahwa permintaan `GET` sederhana sekarang akan mengambil ID versi objek saat ini (121212).

> **Note**
>
> Jika Anda menggunakan permintaan `DeleteObject` yang versinya saat ini adalah penanda hapus (tanpa menentukan ID versi penanda hapus), Amazon S3 tidak akan menghapus penanda hapus, melainkan `PUT`s penanda hapus lainnya.

Untuk menghapus penanda hapus dengan ID versi `NULL`, Anda harus meneruskan `NULL` sebagai ID versi dalam permintaan `DeleteObject`. Gambar berikut menunjukkan bagaimana permintaan `DeleteObject` sederhana dibuat tanpa ID versi yang versinya saat ini adalah penanda hapus, tidak menghapus apa pun, tetapi menambahkan penanda hapus dengan ID versi unik (7498372).



Menggunakan konsol S3

Gunakan langkah-langkah berikut untuk memulihkan objek yang dihapus yang bukan merupakan folder dari bucket S3 Anda, termasuk objek yang ada di dalam folder tersebut.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Dari daftar Bucket, pilih nama bucket yang Anda inginkan.
3. Untuk melihat daftar versi objek di bucket, pilih tombol Buat daftar versi. Anda akan dapat melihat penanda hapus untuk objek yang dihapus.
4. Untuk membatalkan penghapusan objek, Anda harus menghapus penanda hapus ini. Centang kotak di samping penanda hapus objek yang akan dipulihkan, lalu pilih Hapus.
5. Konfirmasi penghapusan pada halaman Hapus objek.
   a. Untuk Hapus objek secara permanen?, masukkan **`permanently delete`**.
   b. Pilih Hapus objek.

> **Note**
>
> Anda tidak dapat menggunakan konsol Amazon S3 untuk membatalkan penghapusan folder. Anda harus menggunakan AWS CLI atau SDK. Sebagai contoh, lihat Bagaimana saya bisa mengambil objek Amazon S3 yang telah dihapus di bucket dengan dukungan Penentuan Versi? di Pusat Pengetahuan AWS .

Penggunaan API REST

Untuk menghapus penanda hapus secara permanen

1. Tetapkan `versionId` ke ID versi ke penanda hapus yang ingin Anda hapus.
2. Kirim permintaan `DELETE Object versionId`.

Example — Menghapus penanda hapus

Contoh berikut menghapus penanda hapus untuk versi `photo.gif` 4857693.

```
DELETE /photo.gif?versionId=4857693 HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Saat Anda menghapus penanda hapus, Amazon S3 menyertakan berikut ini di dalam respons.

```
204 NoContent
x-amz-version-id: versionID
x-amz-delete-marker: true
```

Menggunakan AWS SDK

Untuk informasi tentang menggunakan AWS SDK lain, lihat Pusat AWS Pengembang.

Python

Contoh kode Python berikut menunjukkan cara menghapus penanda hapus dari suatu objek dan dengan demikian menjadikan versi bukan saat ini yang terkini, versi objek saat ini.

```
def revive_object(bucket, object_key):
    """
    Revives a versioned object that was deleted by removing the object's active
    delete marker.
    A versioned object presents as deleted when its latest version is a delete
 marker.
    By removing the delete marker, we make the previous version the latest version
    and the object then presents as *not* deleted.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.
```

```
    :param bucket: The bucket that contains the object.
    :param object_key: The object to revive.
    """
    # Get the latest version for the object.
    response = s3.meta.client.list_object_versions(
        Bucket=bucket.name, Prefix=object_key, MaxKeys=1
    )

    if "DeleteMarkers" in response:
        latest_version = response["DeleteMarkers"][0]
        if latest_version["IsLatest"]:
            logger.info(
                "Object %s was indeed deleted on %s. Let's revive it.",
                object_key,
                latest_version["LastModified"],
            )
            obj = bucket.Object(object_key)
            obj.Version(latest_version["VersionId"]).delete()
            logger.info(
                "Revived %s, active version is now %s  with body '%s'",
                object_key,
                obj.version_id,
                obj.get()["Body"].read(),
            )
        else:
            logger.warning(
                "Delete marker is not the latest version for %s!", object_key
            )
    elif "Versions" in response:
        logger.warning("Got an active version for %s, nothing to do.", object_key)
    else:
        logger.error("Couldn't get any version info for %s.", object_key)
```

Menghapus objek dari bucket dengan dukungan penghapusan MFA

Jika konfigurasi Penentuan Versi bucket didukung penghapusan MFA, pemilik bucket harus menyertakan header permintaan `x-amz-mfa` dalam permintaan untuk secara permanen menghapus versi objek atau mengubah status Penentuan Versi bucket. Permintaan yang mencakup `x-amz-mfa` harus menggunakan HTTPS.

Nilai header adalah rangkaian nomor seri perangkat autentikasi Anda, spasi, dan kode autentikasi yang ditampilkan di atasnya. Jika Anda tidak menyertakan header permintaan ini, permintaan akan gagal.

Untuk informasi selengkapnya tentang perangkat autentikasi, lihat Autentikasi Multi-faktor.

Example — Menghapus objek dari bucket dengan dukungan penghapusan MFA

Contoh berikut menghapus `my-image.jpg` (dengan versi yang ditentukan), yang ada dalam bucket yang dikonfigurasi dengan dukungan penghapusan MFA.

Perhatikan spasi antara *`[SerialNumber]`* dan *`[AuthenticationCode]`*. Untuk informasi selengkapnya, lihat DeleteObject dalam Referensi API Amazon Simple Storage Service.

```
DELETE /my-image.jpg?versionId=3HL4kqCxf3vjVBH40Nrjfkd HTTPS/1.1
Host: bucketName.s3.amazonaws.com
x-amz-mfa: 20899872 301749
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
```

Untuk informasi selengkapnya tentang mengaktifkan penghapusan MFA, lihat Mengonfigurasi penghapusan MFA.

Mengonfigurasi izin objek berversi

Izin untuk objek di Amazon S3 ditetapkan pada tingkat versi. Setiap versi mempunyai pemilik objeknya sendiri. Akun AWS Yang membuat versi objek adalah pemiliknya. Jadi, Anda dapat mengatur izin yang berbeda-beda untuk berbagai versi dari objek yang sama. Untuk melakukannya, Anda harus menentukan ID versi objek yang izinnya ingin Anda tetapkan di permintaan `PUT Object versionId acl`. Untuk penjelasan terperinci dan instruksi tentang penggunaan ACL, lihat Identity and Access Management untuk Amazon S3.

Example — Mengatur izin untuk versi objek

Permintaan berikut menetapkan izin penerima, `BucketOwner@amazon.com`, ke `FULL_CONTROL` pada kunci, `my-image.jpg`, ID versi, `3HL4kqtJvjVBH40Nrjfkd`.

```
PUT /my-image.jpg?acl&versionId=3HL4kqtJvjVBH40Nrjfkd HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
```

```
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU=
Content-Length: 124

<AccessControlPolicy>
  <Owner>
    <ID>75cc57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a</ID>
    <DisplayName>mtd@amazon.com</DisplayName>
  </Owner>
  <AccessControlList>
    <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="CanonicalUser">
        <ID>a9a7b886d6fd24a52fe8ca5bef65f89a64e0193f23000e241bf9b1c61be666e9</ID>
        <DisplayName>BucketOwner@amazon.com</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
  </AccessControlList>
  </AccessControlPolicy>
```

Demikian pula, untuk mendapatkan izin versi objek tertentu, Anda harus menentukan ID versinya di permintaan `GET Object versionId acl`. Anda perlu memasukkan ID versi karena, secara default, `GET Object acl` mengembalikan izin versi objek saat ini.

Example — Mengambil izin untuk versi objek tertentu

Dalam contoh berikut, Amazon S3 mengembalikan izin untuk kunci, `my-image.jpg`, ID versi, `DVBH40Nr8X8gUMLUo`.

```
GET /my-image.jpg?versionId=DVBH40Nr8X8gUMLUo&acl HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:0RQf4/cRonhpaBX5sCYVf1bNRuU
```

Untuk informasi selengkapnya, lihat GetObjectAcl dalam Referensi API Amazon Simple Storage Service.

Bekerja dengan objek dalam bucket dengan Penentuan Versi ditangguhkan

Di Amazon S3, Anda dapat menangguhkan Penentuan Versi untuk berhenti mengumpulkan versi baru dari objek yang sama dalam bucket. Anda mungkin melakukan ini karena Anda hanya

menginginkan satu versi objek dalam bucket. Atau, mungkin Anda tidak ingin menambah biaya untuk beberapa versi.

Saat Anda menangguhkan Penentuan Versi, objek yang ada di bucket tidak berubah. Perubahannya adalah cara Amazon S3 menangani objek di masa mendatang. Topik dalam bagian ini menjelaskan berbagai operasi objek dalam bucket dengan Penentuan Versi ditangguhkan, termasuk menambahkan, mengambil, dan menghapus objek.

Untuk informasi selengkapnya tentang Penentuan Versi S3, lihat Menggunakan Penentuan Versi dalam bucket S3. Untuk informasi selengkapnya tentang pengambilan versi objek, lihat Mengambil versi objek dari bucket dengan dukungan Penentuan Versi.

Topik

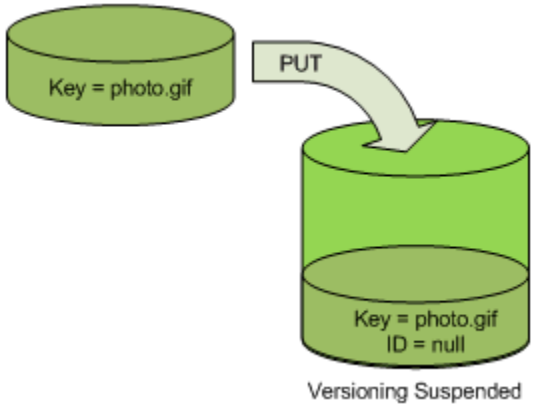
- Menambahkan objek ke bucket dengan Penentuan Versi ditangguhkan
- Mengambil objek dari bucket dengan Penentuan Versi ditangguhkan
- Menghapus objek dari bucket dengan Penentuan Versi ditangguhkan

Menambahkan objek ke bucket dengan Penentuan Versi ditangguhkan

Anda dapat menambahkan objek ke bucket dengan Penentuan Versi ditangguhkan di Amazon S3 untuk membuat objek dengan ID versi null atau menimpa versi objek apa pun dengan ID versi yang cocok.

Setelah Anda menangguhkan Penentuan Versi di bucket, Amazon S3 secara otomatis menambahkan ID versi `null` ke setiap objek berikutnya yang disimpan (menggunakan `PUT`, `POST`, atau `CopyObject`) di bucket tersebut.

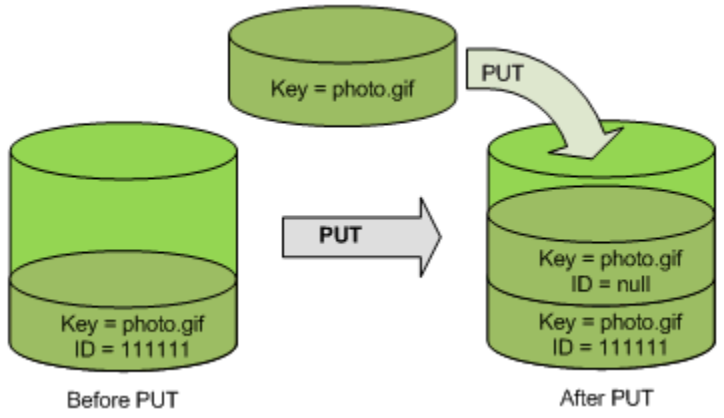
Gambar berikut menunjukkan cara Amazon S3 menambahkan ID versi dari `null` ke objek saat ditambahkan ke bucket dengan Penentuan Versi ditangguhkan.



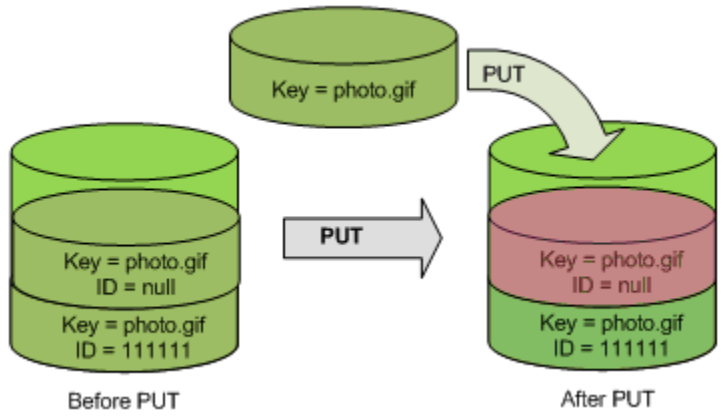
Jika versi null sudah ada dalam bucket dan Anda menambahkan objek lain dengan kunci yang sama, objek tambahan akan menimpa versi null asli.

Apabila terdapat objek berversi dalam bucket, versi yang Anda PUT menjadi versi objek saat ini. Gambar berikut menunjukkan cara menambahkan objek ke bucket yang berisi objek berversi yang tidak menimpa objek yang sudah berada dalam bucket.

Dalam hal ini, versi 111111 sudah ada dalam bucket. Amazon S3 melampirkan ID versi null pada objek yang ditambahkan dan menyimpannya dalam bucket. Versi 111111 tidak ditimpa.



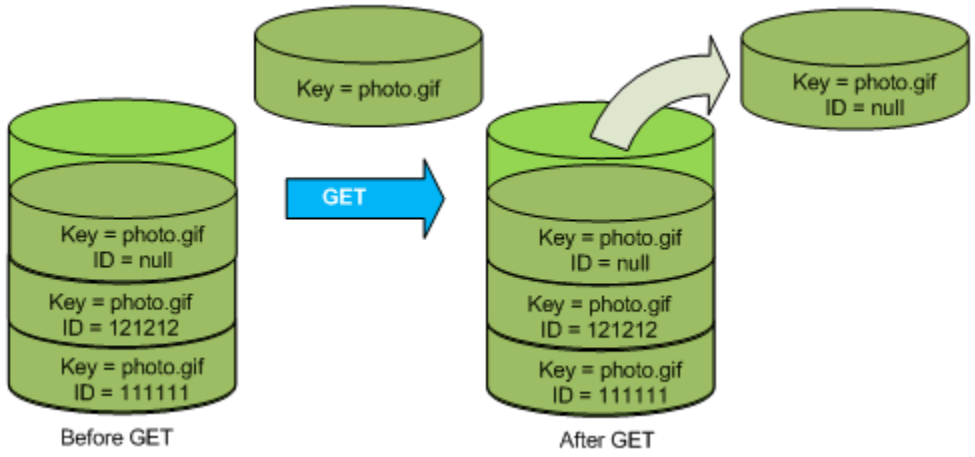
Jika versi null sudah ada di dalam bucket, versi null akan ditimpa, seperti yang ditunjukkan pada gambar berikut.



Meskipun kunci dan ID versi (`null`) dari versi null adalah sama sebelum dan sesudah PUT, isi versi null yang semula disimpan di bucket diganti dengan isi objek PUT ke dalam bucket.

Mengambil objek dari bucket dengan Penentuan Versi ditangguhkan

Permintaan `GET Object` mengembalikan versi objek saat ini, baik apakah Anda telah mengaktifkan Penentuan Versi pada bucket maupun tidak. Gambar berikut menunjukkan cara `GET` sederhana mengembalikan versi objek saat ini.



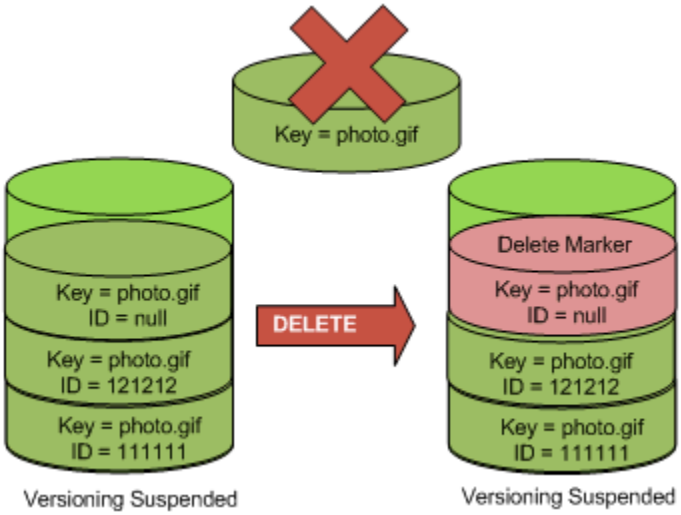
Menghapus objek dari bucket dengan Penentuan Versi ditangguhkan

Anda dapat menghapus objek dari bucket dengan Penentuan Versi ditangguhkan untuk menghapus objek dengan ID versi null.

Jika Penentuan Versi ditangguhkan untuk sebuah bucket, permintaan `DELETE`:

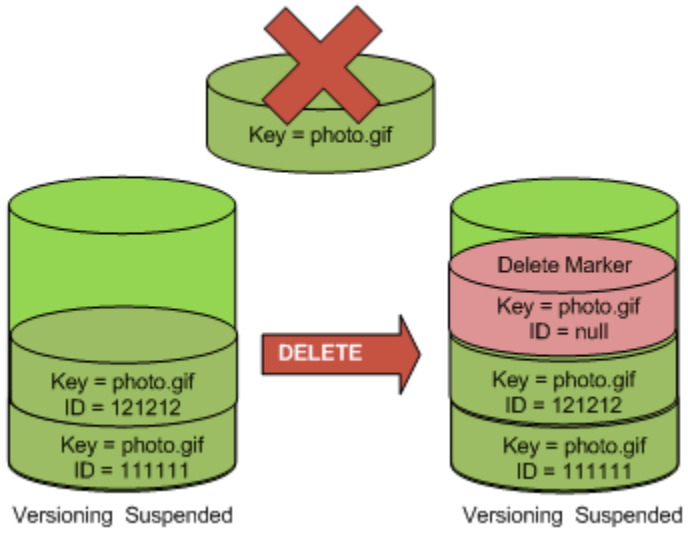
- Hanya dapat menghapus objek yang ID versinya adalah `null`.
- Tidak menghapus apa pun jika tidak ada versi null dari objek dalam bucket.
- Memasukkan penanda hapus ke dalam bucket.

Gambar berikut menunjukkan bagaimana `DELETE` sederhana menghapus versi null. (Permintaan `DELETE` sederhana adalah permintaan yang tidak menentukan ID versi.) Amazon S3 menyisipkan penanda hapus di tempatnya dengan ID versi dari `null`.

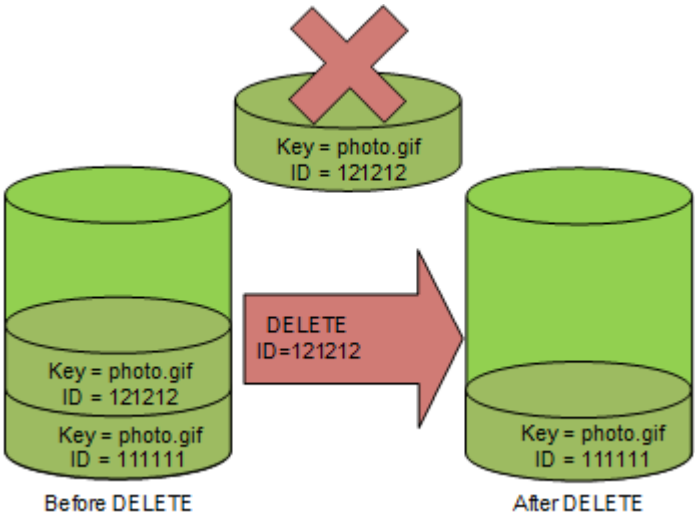


Ingatlah bahwa penanda hapus tidak memiliki konten, sehingga Anda kehilangan konten versi null ketika penanda hapus menggantinya.

Gambar berikut menunjukkan bucket yang tidak memiliki versi null. Dalam kasus ini, DELETE tidak menghapus apa pun; Amazon S3 hanya menyisipkan penanda hapus.



Bahkan di bucket dengan Penentuan Versi ditangguhkan, pemilik bucket dapat menghapus secara permanen versi tertentu dengan menyertakan ID versi dalam permintaan DELETE. Gambar berikut menunjukkan bahwa menghapus versi objek tertentu akan secara permanen menghapus objek tersebut. Hanya pemilik bucket yang dapat menghapus versi objek tertentu.



MenggunakanAWS Backup untuk Amazon S3

Amazon S3 terintegrasi secara native denganAWS Backup, layanan berbasis kebijakan yang dikelola sepenuhnya yang dapat Anda gunakan untuk menentukan kebijakan pencadangan secara terpusat untuk melindungi data Anda di Amazon S3. Setelah Anda menentukan kebijakan pencadangan dan menetapkan sumber daya Amazon S3 ke kebijakan,AWS Backup otomatiskan pembuatan cadangan Amazon S3 dan menyimpan cadangan dengan aman di vault cadangan terenkripsi yang Anda tetapkan dalam paket cadangan Anda.

Saat menggunakanAWS Backup Amazon S3, Anda dapat melakukan tindakan berikut:

- Buat cadangan berkelanjutan dan cadangan berkala. Pencadangan berkelanjutan berguna untuk point-in-time memulihkan, dan cadangan berkala berguna untuk memenuhi kebutuhan retensi data jangka panjang Anda.
- Otomatiskan penjadwalan dan retensi cadangan dengan mengonfigurasi kebijakan pencadangan secara terpusat.
- Pulihkan cadangan data Amazon S3 ke titik waktu yang Anda tentukan.

Selain ituAWS Backup, Anda dapat menggunakan Versi S3 dan Replikasi S3 untuk membantu memulihkan dari penghapusan yang tidak disengaja dan melakukan operasi pemulihan diri Anda sendiri.

Prasyarat

Anda harus mengaktifkan S3 Versioning pada bucket Anda sebelumAWS Backup dapat mencadangkannya.

> **Note**
>
> Sebaiknya tetapkan aturan kedaluwarsa siklus hidup untuk bucket berkemampuan versi yang sedang dicadangkan. Jika Anda tidak menetapkan periode kedaluwarsa siklus hidup, biaya penyimpanan Amazon S3 Anda mungkin meningkat karenaAWS Backup menyimpan semua versi data Amazon S3 Anda.

Mulai

AWS BackupUntuk memulai Amazon S3, lihat Membuat cadangan Amazon S3 di PanduanAWS Backup Pengembang.

Pembatasan dan batasan

Untuk mempelajari batasan, lihat Membuat cadangan Amazon S3 di PanduanAWS Backup Pengembang.

Bekerja dengan objek yang diarsipkan

Untuk mengurangi biaya penyimpanan objek yang jarang diakses, Anda dapat mengarsipkan objek tersebut. Ketika Anda mengarsipkan objek, objek tersebut dipindahkan ke penyimpanan berbiaya rendah, yang berarti Anda tidak dapat mengaksesnya secara real time.

Meskipun objek yang diarsipkan tidak dapat diakses secara waktu nyata, Anda dapat mengembalikannya dalam hitungan menit atau jam, tergantung pada kelas penyimpanan. Anda dapat memulihkan objek yang diarsipkan menggunakan konsol Amazon S3, Operasi Batch S3, REST API, SDK, AWS dan (). AWS Command Line Interface AWS CLI Untuk petunjuk, lihat Memulihkan objek yang diarsipkan.

Objek Amazon S3 di kelas atau tingkatan penyimpanan berikut diarsipkan dan tidak dapat diakses secara waktu nyata:

- Kelas penyimpanan S3 Glacier Flexible Retrieval
- Kelas penyimpanan S3 Glacier Deep Archive
- Tingkat Akses Arsip Intelligent-Tiering S3
- Tingkat Akses Arsip Deep Intelligent-Tiering S3

Untuk memulihkan objek yang diarsipkan, lakukan hal berikut:

- Untuk objek di kelas penyimpanan S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive, Anda harus memulai permintaan pemulihan dan menunggu hingga salinan sementara objek tersedia. Ketika salinan sementara dari objek yang dipulihkan dibuat, kelas penyimpanan objek akan tetap sama. (Permintaan operasi API HeadObject atau GetObject mengembalikan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive sebagai kelas penyimpanan.)
- Untuk objek di tingkat Akses Arsip Intelligent-Tiering S3 dan tingkat Arsip Deep Glacier Intelligent-Tiering S3, Anda harus memulai permintaan pemulihan dan menunggu hingga objek dipindahkan ke tingkat Akses Sering.

Untuk informasi selengkapnya tentang perbandingan semua kelas penyimpanan Amazon S3, lihat Menggunakan kelas penyimpanan Amazon S3. Untuk informasi selengkapnya tentang S3 Intelligent-Tiering, lihat. the section called “Cara kerja S3 Intelligent-Tiering”

Memulihkan objek dari S3 Glacier

Saat Anda menggunakan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, Amazon S3 akan memulihkan salinan sementara objek hanya selama durasi tertentu saja. Setelah itu, layanan akan menghapus salinan objek yang dipulihkan. Anda dapat mengubah masa berlaku salinan yang dipulihkan dengan menerbitkan kembali permintaan pemulihan. Dalam hal ini, Amazon S3 akan memperbarui periode kedaluwarsa relatif terhadap waktu saat ini.

> **Note**
>
> Saat memulihkan objek yang diarsipkan dari S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, Anda perlu membayar objek yang diarsipkan dan salinan yang Anda pulihkan sementara. Untuk informasi tentang harga, lihat Harga Amazon S3.

Memulihkan objek dari S3 Intelligent-Tiering

Saat memulihkan objek dari tingkat Akses Arsip Intelligent-Tiering S3 atau tingkat Akses Arsip Deep Intelligent-Tiering S3, objek akan kembali ke tingkat Akses Sering Intelligent-Tiering S3. Jika objek tidak diakses setelah 30 hari berturut-turut, objek akan secara otomatis berpindah ke tingkat Akses Jarang. Setelah minimal 90 hari berturut-turut tanpa akses, objek akan berpindah ke tingkat Akses Arsip Intelligent-Tiering S3. Jika objek tidak diakses setelah minimal 180 hari berturut-turut, objek akan berpindah ke tingkat Akses Arsip Dalam.

> **Note**
>
> Tidak seperti di kelas penyimpanan S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive, permintaan pemulihan untuk objek S3 Intelligent-Tiering tidak menerima nilai `Days`.

Menggunakan Operasi Batch S3 dengan permintaan pemulihan

Untuk memulihkan lebih dari satu objek Amazon S3 dengan satu permintaan, Anda dapat menggunakan Operasi Batch S3. Anda menyediakan daftar objek yang akan dioperasikan kepada

Operasi Batch S3. Operasi Batch S3 akan memanggil masing-masing operasi API untuk melakukan operasi tertentu. Satu tugas Operasi Batch dapat melakukan operasi tertentu pada miliaran objek yang berisi data sebesar eksabita.

Waktu pemulihan

Amazon S3 menghitung waktu kedaluwarsa salinan objek yang dipulihkan dengan menambahkan jumlah hari yang ditentukan dalam permintaan pemulihan pada saat pemulihan yang diminta telah selesai. Amazon S3 kemudian akan membulatkan waktu yang dihasilkan ke hari berikutnya pada tengah malam Waktu Koordinat Universal (UTC). Sebagai contoh, sebuah salinan objek yang dipulihkan dibuat pada tanggal 15 Oktober 2012, pada pukul 10:30 AM UTC, dan periode pemulihan ditentukan selama 3 hari. Dalam hal ini, salinan yang dipulihkan akan kedaluwarsa pada tanggal 19 Oktober 2012, pukul 00:00 UTC, pada saat itu Amazon S3 akan menghapus salinan objek.

Waktu yang dibutuhkan untuk menyelesaikan pekerjaan pemulihan tergantung pada kelas penyimpanan arsip atau tingkat penyimpanan yang Anda gunakan dan opsi pengambilan yang Anda tentukan: Dipercepat (hanya tersedia untuk S3 Glacier Flexible Retrieval dan Akses Arsip Intelligent-Tiering S3), Standar, atau Massal. Untuk informasi selengkapnya, lihat Opsi pengambilan arsip.

Anda dapat menerima pemberitahuan saat pemulihan telah selesai dengan menggunakan Amazon S3 Event Notifications. Untuk informasi selengkapnya, lihat Notifikasi Peristiwa Amazon S3.

Topik

- Opsi pengambilan arsip
- Memulihkan objek yang diarsipkan

Opsi pengambilan arsip

Berikut ini adalah opsi pengambilan yang tersedia saat memulihkan objek yang diarsipkan dalam Amazon S3:

- Dipercepat-Akses data Anda dengan cepat melalui penyimpanan kelas S3 Glacier Flexible Retrieval atau Akses Arsip Intelligent-Tiering S3. Anda dapat menggunakan opsi ini saat ada permintaan mendesak untuk subset arsip. Untuk semua objek arsip kecuali yang paling besar (250 MB+), data yang diakses dengan menggunakan pengambilan yang dipercepat biasanya akan tersedia dalam waktu 1-5 menit.

> **Note**
>
> Pengambilan yang dipercepat adalah fitur premium dan dibebankan pada tingkat permintaan dan pengambilan yang dipercepat.
> Untuk informasi tentang harga Amazon S3, lihat Harga Amazon S3.

Kapasitas yang disediakan membantu memastikan bahwa kapasitas pengambilan untuk proses pengambilan yang dipercepat dari S3 Glacier Flexible Retrieval selalu tersedia ketika Anda memerlukannya. Untuk informasi selengkapnya, lihat Kapasitas yang disediakan.

- Standar—Akses semua objek yang diarsipkan dalam beberapa jam. Standar adalah opsi default untuk permintaan pengambilan yang tidak menentukan opsi pengambilan. Pengambilan standar biasanya selesai dalam 3–5 jam untuk objek yang disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval atau tingkat Akses Arsip Intelligent-Tiering S3. Pengambilan ini biasanya selesai dalam waktu 12 jam untuk objek yang disimpan di kelas penyimpanan S3 Glacier Deep Archive atau tingkat Akses Arsip Intelligent-Tiering S3. Pengambilan standar gratis untuk objek yang disimpan di S3 Intelligent-Tiering.

> **Note**
>
> - Untuk objek yang disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval atau tingkat S3 Intelligent-Tiering Archive Access, pengambilan Standar yang dimulai dengan menggunakan operasi pemulihan Operasi Batch S3 biasanya dimulai dalam beberapa menit dan selesai dalam 3-5 jam.
> - Untuk objek di kelas penyimpanan S3 Glacier Deep Archive atau tingkat S3 Intelligent-Tiering Deep Archive Access, pengambilan Standar yang dimulai dengan menggunakan operasi pemulihan Operasi Batch biasanya dimulai dalam waktu 9 jam dan selesai dalam 12 jam.

- Massal—Akses data Anda dengan opsi pengambilan biaya terendah di Amazon S3 Glacier. Dengan pengambilan massal, Anda dapat mengambil data dalam jumlah besar, bahkan yang berukuran petabyte, dengan biaya yang murah.

  Untuk objek yang disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval atau tingkat S3 Intelligent-Tiering Archive Access, pengambilan massal biasanya selesai dalam waktu 5—12

jam. Untuk objek yang disimpan di kelas penyimpanan S3 Glacier Deep Archive atau tingkat S3 Intelligent-Tiering Deep Archive Access, pengambilan ini biasanya selesai dalam waktu 48 jam.

Pengambilan massal gratis untuk objek yang disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Intelligent-Tiering.

Tabel berikut merangkum opsi pengambilan arsip. Untuk informasi tentang harga, lihat Harga Amazon S3.

Untuk membuat`Expedited`,`Standard`, atau `Bulk` pengambilan, setel elemen `Tier` permintaan dalam permintaan operasi RestoreObjectREST API ke opsi yang Anda inginkan, atau yang setara di AWS Command Line Interface (AWS CLI) atau AWS SDK. Jika Anda sudah membeli kapasitas yang disediakan, semua pengambilan akan secara otomatis dilayani melalui kapasitas tersebut.

Kapasitas yang disediakan

Kapasitas yang disediakan membantu memastikan bahwa kapasitas pengambilan Anda untuk pengambilan Dipercepat dari S3 Glacier Flexible Retrieval tersedia saat Anda membutuhkannya. Setiap unit kapasitas menyediakan setidaknya tiga pengambilan Dipercepat dapat dilakukan setiap 5 menit, dan memberikan throughput pengambilan hingga 150 megabyte per detik (MBps).

Jika beban kerja Anda membutuhkan akses yang sangat andal dan dapat diprediksi ke sebuah subset data dalam hitungan menit, pertimbangkan untuk membeli kapasitas pengambilan yang disediakan. Tanpa kapasitas yang disediakan, pengambilan Dipercepat mungkin tidak akan diterima jika periode permintaan sedang tinggi. Jika Anda memerlukan akses ke pengambilan Dipercepat dalam keadaan apa pun, sebaiknya beli kapasitas pengambilan yang disediakan.

Unit kapasitas yang disediakan dialokasikan ke a. Akun AWS Dengan demikian, pemohon pengambilan data Dipercepat harus membeli unit kapasitas yang disediakan, bukan pemilik bucket.

Anda dapat membeli kapasitas yang disediakan dengan menggunakan konsol Amazon S3, konsol Amazon S3 Glacier, operasi Purchase Provisioned Capacity REST API, SDK, atau. AWS AWS CLI Untuk informasi harga kapasitas yang disediakan, lihat harga Amazon S3.

Tingkat permintaan inisiasi pemulihan S3 Glacier

Ketika Anda menginisiasi permintaan pemulihan untuk objek yang disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, kuota permintaan pemulihan akan diterapkan untuk file Akun AWS. S3 Glacier mendukung permintaan pemulihan dengan kecepatan

1.000 transaksi per detik. Jika kecepatan ini terlampaui, maka permintaan yang valid akan dibatasi atau ditolak dan Amazon S3 akan mengembalikan kesalahan `ThrottlingException`.

Atau, Anda juga bisa menggunakan Operasi Batch S3 untuk mengambil sejumlah besar objek yang disimpan di S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive dengan satu permintaan. Untuk informasi selengkapnya, lihat Melakukan operasi batch berskala besar pada objek Amazon S3.

Memulihkan objek yang diarsipkan

Objek Amazon S3 di kelas atau tingkatan penyimpanan berikut diarsipkan dan tidak dapat diakses secara waktu nyata:

- Kelas penyimpanan S3 Glacier Flexible Retrieval
- Kelas penyimpanan S3 Glacier Deep Archive
- Tingkat Akses Arsip Intelligent-Tiering S3
- Tingkat Akses Arsip Deep Intelligent-Tiering S3

Objek Amazon S3 yang disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive tidak langsung dapat diakses. Untuk mengakses objek dalam kelas penyimpanan ini, Anda harus mengembalikan salinan sementara objek ke bucket S3 selama durasi tertentu (jumlah hari). Jika Anda menginginkan salinan permanen dari objek tersebut, pulihkan objek, lalu buat salinannya di bucket Amazon S3 Anda. Menyalin objek yang dipulihkan tidak didukung di konsol Amazon S3. Untuk jenis operasi penyalinan ini, gunakan AWS Command Line Interface (AWS CLI), AWS SDK, atau REST API. Kecuali jika Anda membuat salinan dan mengubah kelas penyimpanannya, objek akan tetap disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive. Untuk informasi tentang menggunakan kelas penyimpanan, ini lihat Kelas penyimpanan untuk objek yang jarang diakses.

Untuk mengakses objek di tingkat Akses Arsip Intelligent-Tiering S3 dan Deep Archive Access, Anda harus memulai permintaan pemulihan dan menunggu hingga objek dipindahkan ke tingkat Akses Sering. Saat memulihkan objek dari tingkat Akses Arsip atau tingkat Akses Arsip Dalam, objek akan kembali ke tingkat Akses Sering. Untuk informasi tentang menggunakan kelas penyimpanan, ini lihatKelas penyimpanan untuk mengoptimalkan data secara otomatis dengan pola akses yang berubah atau tidak diketahui.

Untuk informasi umum tentang objek yang diarsipkan, lihat Bekerja dengan objek yang diarsipkan.

> **Note**
>
> - Saat Anda memulihkan objek yang diarsipkan dari kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, Anda membayar objek yang diarsipkan dan salinan yang dipulihkan sementara.
> - Saat Anda memulihkan objek dari S3 Intelligent-Tiering, tidak ada biaya pengambilan untuk pengambilan Standar atau Massal.
> - Permintaan pemulihan berikutnya yang dipanggil pada objek yang diarsipkan yang telah dipulihkan ditagih sebagai `GET` permintaan. Untuk informasi tentang harga, lihat Harga Amazon S3.

Memulihkan objek yang diarsipkan

Anda dapat memulihkan objek yang diarsipkan menggunakan konsol Amazon S3, Amazon S3 REST API, SDK,AWS CLI(), AWS atau Operasi Batch AWS Command Line Interface S3.

Menggunakan konsol S3

Pemulihan objek menggunakan konsol Amazon S3

Gunakan prosedur berikut untuk memulihkan objek yang telah diarsipkan ke kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive Access, atau tingkat penyimpanan Akses Arsip Intelligent-Tiering S3 atau Akses Arsip Dalam.

Untuk memulihkan objek yang diarsipkan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di daftar Bucket pilih nama bucket yang berisi objek yang ingin dipulihkan.
4. Di daftar Objek, pilih objek yang ingin Anda pulihkan, pilih Tindakan, lalu pilih Memulai pemulihan.
5. Jika Anda memulihkan dari S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, masukkan jumlah hari yang Anda inginkan agar data yang diarsipkan dapat diakses di kotak Jumlah hari di mana salinan yang telah dipulihkan tetap tersedia.

6. Untuk Tingkat pengambilan, lakukan salah satu hal berikut:

   - Pilih Pengambilan massal atau Pengambilan standar, lalu pilih Memulai pemulihan.
   - Pilih Pengambilan dipercepat (hanya tersedia untuk S3 Glacier Flexible Retrieval atau Akses Arsip Intelligent-Tiering S3). Jika Anda memulihkan objek di S3 Glacier Flexible Retrieval, Anda dapat memilih apakah akan membeli kapasitas yang disediakan untuk pengambilan Dipercepat. Jika Anda ingin membeli kapasitas yang disediakan, lanjutkan ke langkah berikutnya. Jika tidak, pilih Memulai pemulihan.

   > **Note**
   >
   > Objek dari tingkat S3 Intelligent-Tiering Archive Access dan Deep Archive Access secara otomatis dikembalikan ke tingkat Frequent Access.

7. (Opsional) Jika Anda memulihkan objek di S3 Glacier Flexible Retrieval dan memilih Pengambilan yang dipercepat, Anda dapat memilih apakah akan membeli kapasitas yang disediakan. Kapasitas yang disediakan hanya tersedia untuk objek di S3 Glacier Flexible Retrieval. Jika Anda memiliki kapasitas yang disediakan, pilih Memulai pemulihan untuk memulai pengambilan yang disediakan.

   Jika Anda memiliki kapasitas yang disediakan, semua pengambilan yang Dipercepat dilayani oleh kapasitas yang disediakan. Untuk informasi selengkapnya, lihat Kapasitas yang disediakan.

   - Jika Anda tidak memiliki kapasitas yang disediakan dan tidak ingin membelinya, pilih Memulai pemulihan.
   - Jika Anda tidak memiliki kapasitas yang disediakan, tetapi ingin membeli unit kapasitas yang disediakan (PCU), pilih Beli PCU. Di kotak dialog Beli PCU, pilih jumlah PCU yang ingin Anda beli, konfirmasi pembelian, lalu pilih Beli PCU. Saat Anda mendapatkan pesan Pembelian berhasil, pilih Memulai pemulihan untuk memulai pengambilan yang disediakan.

Menggunakan AWS CLI

Memulihkan objek dari S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive

Contoh berikut menggunakan perintah `restore-object` untuk mengembalikan objek *`dir1/example.obj`* dalam bucket *`example-s3-bucket`* selama 25 hari.

```
aws s3api restore-object --bucket example-s3-bucket --key dir1/example.obj --restore-
request '{"Days":25,"GlacierJobParameters":{"Tier":"Standard"}}'
```

Jika sintaks JSON digunakan dalam contoh hasil dalam kesalahan pada klien Windows, gantikan permintaan pemulihan dengan sintaks berikut:

```
--restore-request Days=25,GlacierJobParameters={"Tier"="Standard"}
```

Memulihkan objek dari S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive

Contoh berikut menggunakan perintah `restore-object` untuk mengembalikan objek *`dir1/example.obj`* dalam bucket *`example-s3-bucket`* ke tingkat Akses Sering.

```
aws s3api restore-object --bucket example-s3-bucket --key dir1/example.obj --restore-
request '{}'
```

> **Note**
>
> Tidak seperti di kelas penyimpanan S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive, permintaan pemulihan untuk objek S3 Intelligent-Tiering tidak menerima nilai `Days`.

Memantau status pemulihan

Untuk memantau status permintaan `restore-object` Anda, gunakan perintah `head-object` berikut:

```
aws s3api head-object --bucket example-s3-bucket --key dir1/example.obj
```

Untuk informasi selengkapnya, lihat restore-object di Referensi Perintah AWS CLI .

Penggunaan API REST

Amazon S3 menyediakan operasi API bagi Anda untuk memulai pemulihan objek yang diarsipkan. Untuk informasi selengkapnya, lihat RestoreObject dalam Referensi API Amazon Simple Storage Service.

Menggunakan AWS SDK

Untuk contoh cara mengembalikan objek yang diarsipkan di S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive dengan SDK, lihat. AWS Gunakan `RestoreObject` dengan AWS SDK atau CLI

Menggunakan Operasi Batch S3

Untuk memulihkan lebih dari satu objek yang diarsipkan dengan satu permintaan, Anda dapat menggunakan Operasi Batch S3. Anda menyediakan daftar objek yang akan dioperasikan kepada Operasi Batch S3. Operasi Batch S3 akan memanggil masing-masing operasi API untuk melakukan operasi tertentu. Satu tugas Operasi Batch dapat melakukan operasi tertentu pada miliaran objek yang berisi data sebesar eksabita.

Untuk membuat tugas Operasi Batch, Anda harus memiliki manifes yang hanya berisi objek yang ingin dipulihkan. Anda dapat membuat manifes dengan menggunakan Inventaris S3, atau Anda dapat memasok file CSV dengan informasi yang diperlukan. Untuk informasi selengkapnya, lihat the section called “Menentukan manifes”.

Sebelum membuat dan menjalankan tugas Operasi Batch S3, Anda harus memberikan izin ke Amazon S3 untuk menjalankan Operasi Batch S3 atas nama Anda. Untuk izin yang diperlukan, lihat the section called “Memberikan izin”.

> **Note**
>
> Tugas Operasi Batch dapat beroperasi baik pada objek kelas penyimpanan S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive atau di objek tingkatan penyimpanan S3 Intelligent-Tiering Archive Access dan Deep Archive Access. Operasi Batch tidak dapat beroperasi pada kedua jenis objek yang diarsipkan dalam tugas yang sama. Untuk memulihkan kedua jenis objek tersebut, Anda harus membuat pekerjaan Operasi Batch terpisah.
>
> Untuk informasi selengkapnya tentang menggunakan Operasi Batch untuk memulihkan objek arsip, lihat the section called “Memulihkan objek”.

Untuk membuat tugas Memulai Operasi Batch Pemulihan Objek S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Operasi Batch.

3. Pilih Buat tugas.
4. Untuk Wilayah AWS, pilih Wilayah tempat Anda ingin membuat tugas Anda.
5. Pada Format manifes, pilih jenis manifes yang akan digunakan.

   - Jika Anda memilih Laporan inventaris S3, masukkan jalur ke objek `manifest.json` yang dihasilkan Amazon S3 sebagai bagian dari laporan inventaris yang diformat CSV. Jika Anda ingin menggunakan versi manifes selain versi terbaru, masukkan ID versi untuk objek `manifest.json`.
   - Jika Anda memilih CSV, masukkan jalur ke objek manifes yang diformat CSV. Objek manifes harus mengikuti format yang dijelaskan di konsol. Jika Anda ingin menggunakan versi selain yang terbaru, Anda dapat secara opsional memasukkan ID versi untuk objek manifes.
6. Pilih Selanjutnya.
7. Di bagian Operasi, pilih Pulihkan.
8. Di bagian Pulihkan, untuk Pulihkan sumber, pilih Glacier Flexible Retrieval atau Glacier Deep Archive atau tingkat Akses Arsip Intelligent-Tiering atau tingkat Akses Arsip Dalam.

   Jika Anda memilih Glacier Flexible Retrieval atau Glacier Deep Archive, masukkan jumlah Jumlah hari tersedianya salinan yang dipulihkan.

   Untuk Tingkat pengambilan, pilih tingkat yang ingin Anda gunakan.
9. Pilih Selanjutnya.
10. Di halaman Konfigurasi opsi tambahan, isilah bagian berikut:

    - Di bagian Opsi tambahan, berikan deskripsi untuk tugas tersebut dan tentukan nomor prioritas untuk tugas tersebut. Angka yang lebih tinggi menunjukkan prioritas yang lebih tinggi. Untuk informasi selengkapnya, lihat the section called “Menetapkan prioritas tugas”.
    - Di bagian Laporan penyelesaian, pilih apakah Operasi Batch harus membuat laporan penyelesaian. Untuk informasi selengkapnya tentang laporan penyelesaian, lihat the section called “Laporan penyelesaian”.
    - Di bagian Izin, Anda harus memberikan izin ke Amazon S3 untuk melakukan Operasi Batch atas nama Anda. Untuk izin yang diperlukan, lihat the section called “Memberikan izin”.
    - (Opsional) Di bagian Tanda tugas, tambahkan tanda dalam pasangan nilai kunci. Untuk informasi selengkapnya, lihat the section called “Menggunakan tanda”.

    Setelah selesai, pilih Berikutnya.

11. Pada halaman Tinjau, verifikasi pengaturan. Jika Anda perlu membuat perubahan, pilih Sebelumnya. Atau, pilih Buat tugas.

Untuk informasi selengkapnya tentang Operasi Batch, lihat Memulihkan objek dengan Operasi Batch dan Membuat pekerjaan Operasi Batch S3.

Memeriksa status pemulihan dan tanggal kedaluwarsa

Anda dapat memeriksa status permintaan pemulihan atau tanggal kedaluwarsa dengan menggunakan konsol Amazon S3, Pemberitahuan Acara Amazon S3, API REST Amazon S3, AWS CLI atau Amazon S3 REST API.

> **Note**
>
> Objek yang dipulihkan dari kelas penyimpanan S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive disimpan hanya untuk jumlah hari yang Anda tentukan. Prosedur berikut mengembalikan tanggal kedaluwarsa untuk salinan ini.
> Objek yang dipulihkan dari S3 Intelligent-Tiering Archive Access dan tingkat penyimpanan Deep Archive Access tidak memiliki tanggal kedaluwarsa dan sebagai gantinya dipindahkan kembali ke tingkat Frequent Access.

Menggunakan konsol S3

Untuk memeriksa status pemulihan dan tanggal kedaluwarsa sebuah objek di konsol Amazon S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di daftar Bucket pilih nama bucket yang berisi objek yang sedang dipulihkan.
4. Dalam daftar Objek, pilih objek yang sedang Anda pulihkan. Halaman detail objek akan muncul.
   - Jika pemulihan belum selesai, di bagian atas halaman, Anda melihat bagian yang mengatakan Pemulihan sedang berlangsung.
   - Jika pemulihan selesai, di bagian atas halaman, Anda akan melihat bagian yang mengatakan Pemulihan selesai. Jika Anda memulihkan dari S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, bagian ini juga menampilkan Tanggal kedaluwarsa pemulihan. Amazon S3 akan menghapus salinan yang dipulihkan dari objek yang diarsipkan pada tanggal ini.

Menggunakan Pemberitahuan Acara Amazon S3

Anda dapat diberi tahu tentang penyelesaian restorasi objek dengan menggunakan `s3:ObjectRestore:Completed` tindakan dengan fitur Pemberitahuan Acara Amazon S3. Untuk informasi selengkapnya tentang mengaktifkan notifikasi peristiwa, lihat Mengaktifkan notifikasi menggunakan Amazon SQS, Amazon SNS, dan. AWS Lambda Untuk informasi selengkapnya tentang berbagai jenis `ObjectRestore` acara, lihatthe section called "Jenis event yang didukung untuk SQS, SNS, dan Lambda".

Menggunakan AWS CLI

Periksa status pemulihan objek dan tanggal kedaluwarsa dengan AWS CLI

Contoh berikut menggunakan perintah `head-object` untuk melihat metadata untuk objek *`dir1/example.obj`* di bucket *`example-s3-bucket`*. Ketika Anda menjalankan perintah ini pada objek yang sedang dipulihkan, Amazon S3 akan menampilkan informasi jika pemulihan sedang berlangsung dan (jika ada) tanggal kedaluwarsanya.

```
aws s3api head-object --bucket example-s3-bucket --key dir1/example.obj
```

Output yang diharapkan (pemulihan sedang berlangsung):

```
{
    "Restore": "ongoing-request=\"true\"",
    "LastModified": "2020-06-16T21:55:22+00:00",
    "ContentLength": 405,
    "ETag": "\"b662d79adeb7c8d787ea7eafb9ef6207\"",
    "VersionId": "wbYaE2vtOV0iIBXrOqGAJt3fP1cHB8Wi",
    "ContentType": "binary/octet-stream",
    "ServerSideEncryption": "AES256",
    "Metadata": {},
    "StorageClass": "GLACIER"
}
```

Output yang diharapkan (pemulihan selesai):

```
{
    "Restore": "ongoing-request=\"false\", expiry-date=\"Wed, 12 Aug 2020 00:00:00 GMT
\"",
    "LastModified": "2020-06-16T21:55:22+00:00",
    "ContentLength": 405,
```

```
    "ETag": "\"b662d79adeb7c8d787ea7eafb9ef6207\"",
    "VersionId": "wbYaE2vtOV0iIBXrOqGAJt3fP1cHB8Wi",
    "ContentType": "binary/octet-stream",
    "ServerSideEncryption": "AES256",
    "Metadata": {},
    "StorageClass": "GLACIER"
}
```

Untuk informasi selengkapnya`head-object`, lihat head-objectdi Referensi AWS CLI Perintah.

Penggunaan API REST

Amazon S3 menyediakan operasi API bagi Anda untuk mengambil metadata objek. Untuk memeriksa status pemulihan dan tanggal kedaluwarsa objek yang diarsipkan menggunakan API REST, lihat HeadObject di Referensi API Amazon Simple Storage Service.

Meningkatkan kecepatan pemulihan yang sedang berlangsung

Anda dapat meningkatkan kecepatan pemulihan saat sedang dalam proses.

Untuk meningkatkan pemulihan yang sedang dalam proses ke tingkat yang lebih cepat

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di daftar Bucket pilih nama bucket yang berisi objek yang ingin dipulihkan.
4. Dalam daftar Objek, pilih objek yang sedang Anda pulihkan. Halaman detail objek akan muncul. Pada halaman detail objek, pilih Tingkatkan tingkat pengambilan. Untuk informasi tentang pemeriksaan status pemulihan objek, lihat Memeriksa status pemulihan dan tanggal kedaluwarsa.
5. Pilih tingkat yang ingin Anda tingkatkan, lalu pilih Memulai pemulihan.

Menggunakan Kunci Objek S3

Kunci Objek S3 dapat membantu Anda mencegah penghapusan atau penimpaan objek Amazon S3 selama jangka waktu tertentu atau tanpa batas waktu. Object Lock menggunakan model write-once-read-many(WORM) untuk menyimpan objek. Anda dapat menggunakan Object Lock untuk membantu memenuhi persyaratan peraturan yang memerlukan penyimpanan WORM, atau untuk menambahkan lapisan perlindungan lain terhadap perubahan atau penghapusan objek.

> **Note**
>
> Kunci Objek S3 telah dinilai oleh Cohasset Associates agar dapat digunakan di lingkungan yang tunduk pada peraturan SEC 17a-4, CFTC, dan FINRA. Untuk informasi selengkapnya tentang keterkaitan Kunci Objek dengan peraturan ini, lihat Penilaian Kepatuhan Cohasset Associates.

Kunci Objek menyediakan dua cara untuk mengelola retensi objek: periode retensi dan penahanan legal. Versi objek dapat memiliki periode retensi, penahanan legal, atau keduanya.

- Periode retensi – Periode retensi menentukan periode waktu tetap selama objek tetap terkunci. Anda dapat mengatur periode retensi unik untuk objek individual. Selain itu, Anda dapat mengatur periode retensi default pada bucket S3. Anda juga dapat membatasi periode retensi minimum dan maksimum yang diijinkan dengan kunci `s3:object-lock-remaining-retention-days` kondisi dalam kebijakan bucket. Ini membantu Anda menetapkan rentang periode retensi dan dengan membatasi periode retensi yang mungkin lebih pendek atau lebih lama dari rentang ini.
- Penahanan legal — Penahanan legal memberikan perlindungan yang sama dengan periode retensi, tetapi tidak memiliki tanggal kedaluwarsa. Penahanan legal akan tetap berlaku sampai Anda menghapusnya secara eksplisit. Penahanan hukum independen dari periode retensi dan ditempatkan pada versi objek individual.

Kunci Objek hanya berfungsi di bucket yang mengaktifkan Versioning S3. Saat Anda mengunci versi objek, Amazon S3 menyimpan informasi kunci di metadata untuk versi objek tersebut. Menempatkan periode retensi atau penahanan legal pada objek hanya melindungi versi yang ditentukan dalam permintaan. Periode retensi dan penahanan hukum tidak mencegah versi baru objek dibuat, atau menghapus penanda yang akan ditambahkan di atas objek. Untuk informasi tentang Versioning S3, lihat Menggunakan Penentuan Versi dalam bucket S3.

Jika Anda memasukkan objek ke dalam bucket yang sudah berisi objek terproteksi dengan nama kunci objek yang sama, Amazon S3 akan membuat versi baru dari objek tersebut. Versi objek yang diproteksi saat ini tetap terkunci sesuai dengan konfigurasi retensinya.

Cara kerja Kunci Objek S3

Topik

- Periode retensi

- Mode retensi
- Penahanan legal
- Praktik terbaik untuk menggunakan S3 Object Lock
- Izin yang diperlukan

Periode retensi

Periode retensi melindungi versi objek untuk jangka waktu tertentu. Saat Anda menempatkan periode retensi pada versi objek, Amazon S3 menyimpan stempel waktu dalam metadata versi objek untuk menunjukkan waktu berakhirnya periode retensi. Setelah periode retensi berakhir, versi objek dapat ditimpa atau dihapus.

Anda dapat menempatkan periode retensi secara eksplisit pada versi objek individual atau pada properti bucket sehingga berlaku untuk semua objek di dalam bucket secara otomatis. Saat Anda menerapkan periode retensi ke versi objek secara eksplisit, Anda menentukan Simpan Hingga Tanggal untuk versi objek. Amazon S3 menyimpan tanggal ini di metadata versi objek.

Anda juga dapat mengatur periode retensi di properti bucket. Saat menetapkan periode retensi pada suatu bucket, Anda menentukan durasi, baik dalam jumlah hari atau tahun, untuk melindungi setiap versi objek yang ditempatkan di dalam bucket tersebut. Saat Anda menempatkan objek di dalam bucket, Amazon S3 menghitung Simpan Hingga Tanggal untuk versi objek dengan menambahkan durasi yang ditentukan ke stempel waktu versioning objek. Kemudian versi objek akan dilindungi persis seperti saat Anda memasang kunci individual dengan periode retensi tersebut pada versi objek.

> **Note**
>
> Jika Anda PUT versi objek yang memiliki mode dan periode retensi individual eksplisit dalam bucket, setelan Kunci Objek individual versi objek akan menggantikan pengaturan retensi properti bucket.

Seperti semua pengaturan Kunci Objek lainnya, periode retensi berlaku untuk versi objek individu. Berbagai versi dari suatu objek dapat memiliki mode dan periode retensi yang berbeda-beda.

Misalkan, Anda memiliki suatu objek yang sudah berada pada periode retensi selama 15 hari dari total 30 hari, dan Anda PUT objek tersebut ke dalam Amazon S3 dengan nama yang sama dan

periode retensi selama 60 hari. Dalam hal ini, permintaan PUT Anda berhasil, dan Amazon S3 akan membuat versi baru objek dengan periode retensi 60 hari. Versi terdahulu akan mempertahankan periode retensi aslinya dan dapat dihapus dalam 15 hari.

Setelah menerapkan pengaturan retensi ke versi objek, Anda dapat memperpanjang periode retensi. Caranya, kirimkan permintaan Kunci Objek baru untuk versi objek dengan Simpan Hingga Tanggal yang lebih lama dari yang saat ini dikonfigurasi untuk versi objek. Amazon S3 menggantikan periode retensi yang ada dengan periode baru yang lebih lama. Setiap pengguna yang mempunyai izin untuk menempatkan periode retensi objek dapat memperpanjang periode retensi untuk versi objek. Untuk menetapkan periode retensi, Anda harus memiliki izin `s3:PutObjectRetention`.

Saat menetapkan periode retensi pada objek atau bucket S3, Anda harus memilih salah satu dari dua mode retensi: kepatuhan atau tata kelola.

Mode retensi

Kunci Objek S3 menyediakan dua mode retensi yang menerapkan tingkat perlindungan berbeda pada objek Anda:

- Mode kepatuhan
- Mode tata kelola

Di mode kepatuhan, versi objek terlindungi tidak dapat ditimpa atau dihapus oleh pengguna mana pun, termasuk pengguna root di Akun AWS Anda. Ketika sebuah objek dikunci dalam mode kepatuhan, mode retensinya tidak dapat diubah, dan periode retensinya tidak dapat disingkat. Mode kepatuhan membantu memastikan bahwa versi objek tidak dapat ditimpa atau dihapus selama periode retensi.

> **Note**
>
> Satu-satunya cara untuk menghapus objek di bawah mode kepatuhan sebelum tanggal retensi berakhir adalah dengan menghapus yang terkait Akun AWS.

Di mode tata kelola, pengguna tidak dapat menimpa atau menghapus versi objek atau mengubah pengaturan kuncinya kecuali memiliki izin khusus. Dengan mode tata kelola, Anda melindungi objek agar tidak dihapus oleh sebagian besar pengguna, tetapi Anda masih dapat memberikan izin

kepada beberapa pengguna untuk mengubah pengaturan penyimpanan atau menghapus objek jika diperlukan. Anda juga dapat menggunakan mode tata kelola untuk menguji pengaturan periode retensi sebelum membuat periode retensi mode kepatuhan.

Untuk menimpa atau menghapus pengaturan retensi mode tata kelola, Anda harus memiliki izin `s3:BypassGovernanceRetention` dan harus secara eksplisit menyertakan `x-amz-bypass-governance-retention:true` sebagai header permintaan dengan permintaan apa pun yang memerlukan penimpaan mode tata kelola.

> **Note**
>
> Secara default, konsol Amazon S3 menyertakan header `x-amz-bypass-governance-retention:true`. Jika Anda mencoba menghapus objek yang dilindungi oleh mode tata kelola dan memiliki izin `s3:BypassGovernanceRetention`, operasi akan berhasil.

Penahanan legal

Dengan Kunci Objek Anda juga dapat menempatkan penahanan legal pada versi objek. Seperti periode retensi, penahanan legal mencegah penimpaan atau penghapusan versi objek. Namun, penahanan legal tidak memiliki jangka waktu tertentu dan tetap berlaku sampai penahanan tersebut dihapus. Penahanan legal dapat ditempatkan dan dihapus dengan bebas oleh pengguna mana pun yang memiliki izin `s3:PutObjectLegalHold`.

Penahanan legal terpisah dari periode retensi. Menempatkan penahanan legal pada versi objek tidak memengaruhi mode retensi atau periode retensi untuk versi objek tersebut.

Misalnya, Anda menempatkan penahanan legal pada versi objek dan versi objek juga dilindungi oleh periode retensi. Jika periode retensi berakhir, objek tidak kehilangan perlindungan WORM. Bahkan, penahanan legal akan terus melindungi objek hingga pengguna yang berwenang secara eksplisit menghapus penahanan legal. Demikian halnya, jika Anda menghapus penahanan legal saat periode retensi versi objek masih berlaku, versi objek akan tetap terlindungi hingga periode retensi berakhir.

Praktik terbaik untuk menggunakan S3 Object Lock

Pertimbangkan untuk menggunakan mode Tata Kelola jika Anda ingin melindungi objek agar tidak dihapus oleh sebagian besar pengguna selama periode retensi yang telah ditentukan sebelumnya, tetapi pada saat yang sama ingin beberapa pengguna dengan izin khusus memiliki fleksibilitas untuk mengubah pengaturan retensi atau menghapus objek.

Pertimbangkan untuk menggunakan mode Kepatuhan jika Anda tidak ingin pengguna mana pun, termasuk pengguna root di Anda Akun AWS, dapat menghapus objek selama periode retensi yang telah ditentukan sebelumnya. Anda dapat menggunakan mode ini jika Anda memiliki persyaratan untuk menyimpan data yang sesuai.

Anda dapat menggunakan Legal Hold ketika Anda tidak yakin berapa lama Anda ingin objek Anda tetap tidak berubah. Ini bisa jadi karena Anda memiliki audit eksternal data Anda yang akan datang dan ingin menjaga objek tetap tidak berubah sampai audit selesai. Sebagai alternatif, Anda mungkin memiliki proyek yang sedang berlangsung menggunakan kumpulan data yang ingin Anda pertahankan hingga proyek selesai.

Izin yang diperlukan

Operasi Kunci Objek memerlukan izin khusus. Tergantung pada operasi yang sedang dijalankan, Anda mungkin memerlukan salah satu izin berikut ini:

- `s3:BypassGovernanceRetention`
- `s3:GetBucketObjectLockConfiguration`
- `s3:GetObjectLegalHold`
- `s3:GetObjectRetention`
- `s3:PutBucketObjectLockConfiguration`
- `s3:PutObjectLegalHold`
- `s3:PutObjectRetention`

Untuk daftar lengkap izin Amazon S3 dengan deskripsi, lihat Tindakan, sumber daya, dan kunci kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Untuk informasi tentang menggunakan ketentuan dengan izin, lihat Contoh kebijakan bucket menggunakan tombol kondisi.

Pertimbangan Kunci Objek

Kunci Objek Amazon S3 dapat membantu mencegah penghapusan atau penimpaan objek selama jangka waktu tertentu atau tanpa batas waktu.

Anda dapat menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), AWS SDK, atau Amazon S3 REST API untuk melihat atau menyetel informasi Kunci Objek. Untuk informasi umum tentang kemampuan Kunci Objek S3, lihat Menggunakan Kunci Objek S3.

> **Important**
>
> - Setelah mengaktifkan Kunci Objek pada bucket, Anda tidak dapat menonaktifkan Kunci Objek atau menangguhkan Penentuan Versi untuk bucket tersebut.
> - Bucket S3 dengan Kunci Objek tidak dapat digunakan sebagai bucket tujuan untuk log akses server. Untuk informasi selengkapnya, lihat the section called "Pencatatan akses server".

Topik

- Izin untuk melihat informasi kunci
- Melewati mode tata kelola
- Menggunakan Kunci Objek dengan Replikasi S3
- Menggunakan Kunci Objek dengan Inventaris Amazon S3
- Mengelola kebijakan Siklus Hidup S3 dengan Object Lock
- Mengelola penanda hapus dengan Object Lock
- Menggunakan Lensa Penyimpanan S3 dengan Kunci Objek
- Mengunggah objek ke bucket yang diaktifkan Object Lock
- Mengonfigurasi peristiwa dan pemberitahuan
- Menetapkan batas periode retensi dengan kebijakan bucket

Izin untuk melihat informasi kunci

Anda dapat melihat status Kunci Objek dari versi objek Amazon S3 secara terprogram dengan menggunakan operasi HeadObject atau GetObject. Kedua operasi tersebut akan mengembalikan mode retensi, simpan hingga tanggal, dan status penahanan legal untuk versi objek tertentu. Selain itu, Anda dapat melihat status Object Lock untuk beberapa objek di bucket S3 menggunakan S3 Inventory.

Untuk melihat mode retensi dan periode retensi versi objek, Anda harus memiliki izin `s3:GetObjectRetention`. Untuk melihat status kontrol legal versi objek, Anda harus memiliki izin `s3:GetObjectLegalHold`. Untuk melihat konfigurasi retensi default bucket, Anda harus memiliki izin `s3:GetBucketObjectLockConfiguration`. Jika Anda membuat permintaan untuk

konfigurasi Kunci Objek pada bucket yang tidak memiliki Kunci Objek S3 yang aktif, Amazon S3 akan memberikan pesan kesalahan.

Melewati mode tata kelola

Jika Anda memiliki `s3:BypassGovernanceRetention` izin, Anda dapat melakukan operasi pada versi objek yang terkunci dalam mode tata kelola seolah-olah objek tersebut tidak dilindungi. Operasi ini termasuk menghapus versi objek, memperpendek periode retensi, atau menghapus periode retensi Kunci Objek dengan menempatkan permintaan `PutObjectRetention` baru dengan parameter kosong.

Untuk melewati mode tata kelola, Anda harus secara eksplisit menunjukkan permintaan bahwa Anda ingin melewati mode ini. Untuk melakukannya, sertakan `x-amz-bypass-governance-retention:true` header dengan permintaan operasi `PutObjectRetention` API Anda, atau gunakan parameter yang setara dengan permintaan yang dibuat melalui AWS CLI atau AWS SDK. Konsol S3 secara otomatis menerapkan header ini untuk permintaan yang dibuat melalui konsol S3 jika Anda memiliki izin `s3:BypassGovernanceRetention`.

> **Note**
>
> Melewati mode tata kelola tidak akan memengaruhi status penahanan legal versi objek. Jika versi objek memiliki penahanan legal yang aktif, penahanan legal tersebut tetap berlaku dan mencegah permintaan untuk menimpa atau menghapus versi objek.

Menggunakan Kunci Objek dengan Replikasi S3

Anda dapat menggunakan Kunci Objek dengan Replikasi S3 untuk mengaktifkan penyalinan otomatis dan asinkron dari objek terkunci dan metadata retensinya, di seluruh bucket S3. Ini berarti bahwa untuk objek yang direplikasi, Amazon S3 mengambil konfigurasi kunci objek dari bucket sumber. Dengan kata lain, jika bucket sumber mengaktifkan Object Lock, bucket tujuan juga harus mengaktifkan Object Lock. Jika objek langsung diunggah ke bucket tujuan (di luar S3 Replication), objek tersebut akan diatur Object Lock pada bucket tujuan. Saat Anda menggunakan replikasi, objek di bucket sumber direplikasi ke satu atau lebih bucket tujuan.

Untuk menyiapkan replikasi pada bucket dengan Object Lock diaktifkan, Anda dapat menggunakan konsol S3, Amazon S3 REST API AWS CLI, atau SDK. AWS

> **Note**
>
> Untuk menggunakan Object Lock dengan replikasi, Anda harus memberikan dua izin tambahan pada bucket S3 sumber dalam peran AWS Identity and Access Management (IAM) yang Anda gunakan untuk mengatur replikasi. Dua izin tambahan tersebut adalah `s3:GetObjectRetention` dan`s3:GetObjectLegalHold`. Jika peran tersebut memiliki pernyataan izin `s3:Get*`, pernyataan itu memenuhi persyaratan. Untuk informasi selengkapnya, lihat Menyiapkan izin untuk replikasi langsung.
> Untuk informasi umum tentang Replikasi S3, lihat. Mereplikasi ikhtisar objek
> Untuk contoh pengaturan Replikasi S3, lihat Contoh untuk mengonfigurasi replikasi langsung.

Menggunakan Kunci Objek dengan Inventaris Amazon S3

Anda dapat mengonfigurasi Inventaris Amazon S3 untuk membuat daftar objek di bucket S3 sesuai jadwal yang sudah ditentukan. Anda dapat mengonfigurasi Inventaris Amazon S3 untuk menyertakan metadata Kunci Objek berikut untuk objek Anda:

- Simpan hingga tanggal
- Mode retensi
- Status penahanan legal

Untuk informasi selengkapnya, lihat Inventaris Amazon S3.

Mengelola kebijakan Siklus Hidup S3 dengan Object Lock

Konfigurasi manajemen siklus aktif objek akan tetap berfungsi secara normal pada objek yang terlindungi, termasuk menempatkan penanda hapus. Namun, versi objek yang terkunci tidak dapat dihapus oleh kebijakan kedaluwarsa Siklus Hidup S3. Object Lock dipertahankan terlepas dari kelas penyimpanan mana objek berada di dan sepanjang transisi Siklus Hidup S3 antar kelas penyimpanan.

Untuk informasi selengkapnya tentang mengelola siklus hidup objek, lihat Mengelola siklus hidup penyimpanan Anda.

Mengelola penanda hapus dengan Object Lock

Meskipun tidak dapat menghapus versi objek yang dilindungi, Anda tetap dapat membuat penanda hapus untuk objek tersebut. Memasang penanda hapus pada sebuah objek tidak akan menghapus objek atau versi objeknya. Namun, hal ini membuat Amazon S3 bertindak seolah-olah objek tersebut telah dihapus. Untuk informasi selengkapnya, lihat Bekerja dengan penanda hapus.

> **Note**
>
> Penanda hapus tidak dilindungi WORM, terlepas dari periode retensi atau kontrol legal yang diterapkan pada objek yang mendasarinya.

Menggunakan Lensa Penyimpanan S3 dengan Kunci Objek

untuk melihat metrik terkait jumlah byte penyimpanan yang diaktifkan oleh Kunci Objek dan jumlah objek, Anda dapat menggunakan Lensa Penyimpanan Amazon S3. Lensa Penyimpanan S3 adalah fitur analisis penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan objek.

Untuk informasi selengkapnya, lihat Menggunakan Lensa Penyimpanan S3 untuk melindungi data Anda.

Untuk daftar lengkap metrik, lihat Glosarium metrik Amazon S3 Storage Lens.

Mengunggah objek ke bucket yang diaktifkan Object Lock

`Content-MD5`Header diperlukan untuk setiap permintaan untuk mengunggah objek dengan periode retensi yang dikonfigurasi menggunakan Object Lock. Intisari MD5 adalah cara untuk memverifikasi integritas objek Anda setelah mengunggahnya ke ember. Setelah mengunggah objek, Amazon S3 menghitung intisari MD5 objek dan membandingkannya dengan nilai yang Anda berikan. Permintaan hanya berhasil jika kedua intisari cocok. Konsol S3 secara otomatis menambahkan header ini, namun Anda harus menentukan header ini saat menggunakan PutObjectAPI.

Untuk informasi selengkapnya, lihat Menggunakan Content-MD5 saat mengunggah objek.

Mengonfigurasi peristiwa dan pemberitahuan

Anda dapat menggunakan Pemberitahuan Acara Amazon S3 untuk melacak akses dan perubahan pada konfigurasi dan data Object Lock Anda dengan menggunakan. AWS CloudTrail Untuk informasi tentang CloudTrail, lihat Apa itu AWS CloudTrail? dalam AWS CloudTrail User Guide.

Anda juga dapat menggunakan Amazon CloudWatch untuk menghasilkan peringatan berdasarkan data ini. Untuk informasi tentang CloudWatch, lihat Apa itu Amazon CloudWatch? di Panduan CloudWatch Pengguna Amazon.

Menetapkan batas periode retensi dengan kebijakan bucket

Anda dapat menetapkan periode retensi minimum dan maksimum yang diizinkan untuk bucket menggunakan kebijakan bucket. Periode retensi maksimum adalah 100 tahun.

Contoh berikut menunjukkan kebijakan bucket yang menggunakan kunci kondisi `s3:object-lock-remaining-retention-days` untuk menetapkan periode retensi maksimum selama 10 hari.

```
{
    "Version": "2012-10-17",
    "Id": "SetRetentionLimits",
    "Statement": [
        {
            "Sid": "SetRetentionPeriod",
            "Effect": "Deny",
            "Principal": "*",
            "Action": [
                "s3:PutObjectRetention"
            ],
            "Resource": "arn:aws:s3:::example-s3-bucket1/*",
            "Condition": {
                "NumericGreaterThan": {
                    "s3:object-lock-remaining-retention-days": "10"
                }
            }
        }
    ]
}
```

> **Note**
>
> Jika bucket Anda adalah bucket tujuan konfigurasi replikasi, Anda dapat mengatur periode retensi minimum dan maksimum yang diizinkan untuk replika objek yang dibuat dengan menggunakan replikasi. Untuk melakukannya, Anda harus mengizinkan tindakan `s3:ReplicateObject` dalam kebijakan bucket. Untuk informasi selengkapnya tentang izin replikasi, lihat the section called “Menyiapkan izin”.

Untuk informasi selengkapnya tentang kebijakan bucket, lihat topik berikut:

- Kunci tindakan, sumber daya, dan kondisi untuk Amazon S3 di Referensi Otorisasi Layanan
- Operasi objek
- Contoh kebijakan bucket menggunakan tombol kondisi

Mengonfigurasi Kunci Objek S3

Dengan Amazon S3 Object Lock, Anda dapat menyimpan objek di Amazon S3 dengan menggunakan write-once-read-manymodel (WORM). Anda dapat menggunakan Kunci Objek S3 untuk mencegah objek terhapus atau ditimpa selama jangka waktu tertentu atau tanpa batas waktu. Untuk informasi umum tentang kemampuan Kunci Objek, lihat Menggunakan Kunci Objek S3.

Sebelum mengunci objek apa pun, Anda harus mengaktifkan Penentuan Versi S3 dan Kunci Objek pada bucket. Setelah itu, Anda dapat mengatur periode retensi, penyimpanan yang sah, atau keduanya.

Untuk bekerja dengan Kunci Objek, Anda harus memiliki izin tertentu. Untuk daftar izin yang terkait dengan berbagai operasi Kunci Objek, lihat the section called “Izin yang diperlukan”.

> **Important**
>
> - Setelah mengaktifkan Kunci Objek pada bucket, Anda tidak dapat menonaktifkan Kunci Objek atau menangguhkan Penentuan Versi untuk bucket tersebut.
> - Bucket S3 dengan Kunci Objek tidak dapat digunakan sebagai bucket tujuan untuk log akses server. Untuk informasi selengkapnya, lihat the section called “Pencatatan akses server”.

Topik

- Aktifkan Kunci Objek saat membuat bucket S3 yang baru
- Aktifkan Kunci Objek pada bucket S3 yang ada
- Mengatur atau memodifikasi penyimpanan yang sah pada objek S3
- Menetapkan atau memodifikasi periode retensi pada objek S3
- Menetapkan atau memodifikasi periode retensi default pada bucket S3

Aktifkan Kunci Objek saat membuat bucket S3 yang baru

Anda dapat mengaktifkan Object Lock saat membuat bucket S3 baru dengan menggunakan konsol Amazon S3 AWS Command Line Interface ,AWS CLI(), SDK AWS , atau Amazon S3 REST API.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih Buat bucket.

   Halaman Buat bucket terbuka.
4. Untuk Nama bucket, masukkan nama untuk bucket Anda.

   > **Note**
   >
   > Setelah membuat bucket, Anda tidak dapat mengubah namanya. Untuk informasi selengkapnya tentang penamaan bucket, lihat Peraturan penamaan bucket.

5. Untuk Wilayah, pilih Wilayah AWS tempat Anda ingin ember berada.
6. Di bawah Kepemilikan Objek, pilih untuk menonaktifkan atau mengaktifkan daftar kontrol akses (ACL) dan mengontrol kepemilikan objek yang diunggah di bucket Anda.
7. Di bawah Pengaturan Blokir Akses Publik untuk bucket ini, pilih pengaturan Blokir Akses Publik yang ingin Anda terapkan ke bucket.
8. Di bawah Penentuan Versi Bucket, pilih Diaktifkan.

   Kunci Objek hanya berfungsi dengan bucket berversi.
9. (Opsional) Di bawah Tanda, Anda dapat memilih untuk menambahkan tanda ke bucket Anda. Tanda adalah pasangan nilai kunci yang digunakan untuk mengkategorikan penyimpanan dan mengalokasikan biaya.
10. Di bawah Pengaturan lanjutan, cari Kunci Objek dan pilih Aktifkan.

    Anda harus memahami bahwa mengaktifkan Kunci Objek akan membuat objek di dalam bucket ini terkunci secara permanen.
11. Pilih Buat bucket.

Menggunakan AWS CLI

Contoh `create-bucket` berikut membuat bucket S3 baru bernama *`example-s3-bucket1`* dengan Kunci Objek yang diaktifkan:

```
aws s3api create-bucket --bucket example-s3-bucket1 --object-lock-enabled-for-bucket
```

Untuk informasi dan contoh selengkapnya, lihat create-bucket di Referensi Perintah AWS CLI .

> **Note**
>
> Anda dapat menjalankan AWS CLI perintah dari konsol dengan menggunakan AWS CloudShell. AWS CloudShell adalah shell pra-otentikasi berbasis browser yang dapat Anda luncurkan langsung dari file. AWS Management Console Untuk informasi lebih lanjut, lihat Apa itu CloudShell? dalam AWS CloudShell User Guide.

Penggunaan API REST

Anda dapat menggunakan API REST untuk membuat bucket S3 baru dengan Kunci Objek yang diaktifkan. Untuk informasi selengkapnya, lihat CreateBucket di Referensi API Amazon Simple Storage Service.

Menggunakan AWS SDK

Untuk contoh cara mengaktifkan Object Lock saat membuat bucket S3 baru dengan AWS SDK, lihat. Gunakan `CreateBucket` dengan AWS SDK atau CLI

Untuk contoh cara mendapatkan konfigurasi Object Lock saat ini dengan AWS SDK, lihatGunakan `GetObjectLockConfiguration` dengan AWS SDK atau CLI.

Untuk skenario interaktif yang mendemonstrasikan fitur Object Lock yang berbeda menggunakan AWS SDK, lihat. Bekerja dengan fitur kunci objek Amazon S3 menggunakan SDK AWS

Untuk informasi umum tentang penggunaan AWS SDK yang berbeda, lihatMengembangkan dengan Amazon S3 menggunakan SDK AWS.

Aktifkan Kunci Objek pada bucket S3 yang ada

Anda dapat mengaktifkan Object Lock untuk bucket S3 yang ada dengan menggunakan konsol Amazon S3, SDK AWS , AWS CLI atau Amazon S3 REST API.

Menggunakan konsol S3

> **Note**
>
> Kunci Objek hanya berfungsi dengan bucket berversi.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di daftar Bucket, pilih nama bucket yang ingin Anda aktifkan Kunci Objek.
4. Pilih tab Properti.
5. Di bawah Properti, gulir ke bawah ke bagian Kunci Objek, lalu pilih Edit.
6. Di bawah Kunci Objek, pilih Aktifkan.

   Anda harus memahami bahwa mengaktifkan Kunci Objek akan membuat objek di dalam bucket ini terkunci secara permanen.
7. Pilih Simpan perubahan.

Menggunakan AWS CLI

`put-object-lock-configuration`Contoh perintah berikut menetapkan periode retensi Kunci Objek 50 hari pada bucket bernama *`example-s3-bucket1`*:

```
aws s3api put-object-lock-configuration --bucket example-s3-bucket1 --object-lock-
configuration='{ "ObjectLockEnabled": "Enabled", "Rule": { "DefaultRetention":
 { "Mode": "COMPLIANCE", "Days": 50 }}}'
```

Untuk informasi dan contoh selengkapnya, lihat put-object-lock-configuration di Referensi Perintah AWS CLI .

> **Note**
>
> Anda dapat menjalankan AWS CLI perintah dari konsol dengan menggunakan AWS CloudShell. AWS CloudShell adalah shell pra-otentikasi berbasis browser yang dapat Anda

luncurkan langsung dari file. AWS Management Console Untuk informasi lebih lanjut, lihat Apa itu CloudShell? dalam AWS CloudShell User Guide.

Penggunaan API REST

Anda dapat menggunakan API REST Amazon S3 untuk mengaktifkan Kunci Objek pada bucket S3 yang sudah ada. Untuk informasi selengkapnya, lihat PutObjectLockConfiguration di Referensi API Amazon Simple Storage Service.

Menggunakan AWS SDK

Untuk contoh cara mengaktifkan Object Lock untuk bucket S3 yang ada dengan AWS SDK, lihat. Gunakan `PutObjectLockConfiguration` dengan AWS SDK atau CLI

Untuk contoh cara mendapatkan konfigurasi Object Lock saat ini dengan AWS SDK, lihatGunakan `GetObjectLockConfiguration` dengan AWS SDK atau CLI.

Untuk skenario interaktif yang mendemonstrasikan fitur Object Lock yang berbeda menggunakan AWS SDK, lihat. Bekerja dengan fitur kunci objek Amazon S3 menggunakan SDK AWS

Untuk informasi umum tentang penggunaan AWS SDK yang berbeda, lihatMengembangkan dengan Amazon S3 menggunakan SDK AWS.

Mengatur atau memodifikasi penyimpanan yang sah pada objek S3

Anda dapat menyetel atau menghapus penahanan hukum pada objek S3 dengan menggunakan konsol Amazon S3, SDK AWS CLI AWS , atau Amazon S3 REST API.

Important

- Jika Anda ingin mengatur penyimpanan yang sah pada suatu objek, bucket objek harus sudah mengaktifkan Kunci Objek.
- Jika Anda memiliki versi objek PUT yang memiliki mode dan periode retensi individual eksplisit dalam bucket, pengaturan Kunci Objek individual versi objek akan menimpa pengaturan retensi properti bucket apa pun.

Untuk informasi selengkapnya, lihat the section called “Penahanan legal”.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di daftar Bucket, pilih nama bucket yang berisi objek yang ingin Anda tetapkan atau ubah penyimpanan sahnya.
4. Dalam daftar Objek, pilih objek yang ingin Anda atur atau ubah penyimpanan sahnya.
5. Pada halaman Properti objek, temukan bagian Penyimpanan yang Sah Kunci Objek, lalu pilih Edit.
6. Pilih Aktifkan untuk menetapkan penyimpanan yang sah atau Nonaktifkan untuk menghapus penyimpanan yang sah.
7. Pilih Simpan perubahan.

Menggunakan AWS CLI

Contoh `put-object-legal-hold` berikut menetapkan penyimpanan yang sah pada objek *my-image.fs* dalam bucket bernama *example-s3-bucket1*:

```
aws s3api put-object-legal-hold --bucket example-s3-bucket1 --key my-image.fs --legal-hold="Status=ON"
```

Contoh `put-object-legal-hold` berikut menghapus penyimpanan yang sah pada objek *my-image.fs* dalam bucket bernama *example-s3-bucket1*:

```
aws s3api put-object-legal-hold --bucket example-s3-bucket1 --key my-image.fs --legal-hold="Status=OFF"
```

Untuk informasi dan contoh selengkapnya, lihat put-object-legal-hold di Referensi Perintah AWS CLI .

> **Note**
>
> Anda dapat menjalankan AWS CLI perintah dari konsol dengan menggunakan AWS CloudShell. AWS CloudShell adalah shell pra-otentikasi berbasis browser yang dapat Anda luncurkan langsung dari file. AWS Management Console Untuk informasi lebih lanjut, lihat Apa itu CloudShell? dalam AWS CloudShell User Guide.

Penggunaan API REST

Anda dapat menggunakan API REST untuk menetapkan atau memodifikasi penyimpanan yang sah pada objek. Untuk informasi selengkapnya, lihat PutObjectLegalHold di Referensi API Amazon Simple Storage Service.

Menggunakan AWS SDK

Untuk contoh cara mengatur penahanan hukum pada objek dengan AWS SDK, lihatGunakan `PutObjectLegalHold` dengan AWS SDK atau CLI.

Untuk contoh cara mendapatkan status penahanan hukum saat ini dengan AWS SDK, lihatDapatkan konfigurasi penahanan hukum objek Amazon S3 menggunakan SDK AWS.

Untuk skenario interaktif yang mendemonstrasikan fitur Object Lock yang berbeda menggunakan AWS SDK, lihat. Bekerja dengan fitur kunci objek Amazon S3 menggunakan SDK AWS

Untuk informasi umum tentang penggunaan AWS SDK yang berbeda, lihatMengembangkan dengan Amazon S3 menggunakan SDK AWS.

Menetapkan atau memodifikasi periode retensi pada objek S3

Anda dapat menyetel atau mengubah periode retensi pada objek S3 menggunakan konsol Amazon S3, SDK AWS CLI AWS , atau Amazon S3 REST API.

> **Important**
>
> - Jika Anda ingin mengatur periode retensi pada objek, bucket objek harus sudah mengaktifkan Kunci Objek.
> - Jika Anda memiliki versi objek PUT yang memiliki mode dan periode retensi individual eksplisit dalam bucket, pengaturan Kunci Objek individual versi objek akan menimpa pengaturan retensi properti bucket apa pun.
> - Satu-satunya cara untuk menghapus objek di bawah mode kepatuhan sebelum tanggal retensi berakhir adalah dengan menghapus yang terkait Akun AWS.

Untuk informasi selengkapnya, lihat Periode retensi.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di daftar Bucket, pilih nama bucket yang berisi objek yang ingin Anda tetapkan atau ubah periode retensinya.
4. Dalam daftar Objek, pilih objek yang ingin Anda tetapkan atau ubah periode retensinya.
5. Pada halaman Properti object, temukan bagian Retensi Kunci Objek, lalu pilih Edit.
6. Di bawah Retensi, pilih Aktifkan untuk menetapkan periode retensi atau Nonaktifkan untuk menghapus periode retensi.
7. Jika Anda memilih Aktifkan, di bawah Mode retensi, pilih mode Tata Kelola atau mode Kepatuhan. Untuk informasi selengkapnya, lihat Mode retensi.
8. Di bawah Pertahankan hingga tanggal, pilih tanggal yang Anda inginkan untuk mengakhiri periode retensi. Selama periode ini, objek Anda dilindungi WORM dan tidak dapat ditimpa atau dihapus. Untuk informasi selengkapnya, lihat Periode retensi.
9. Pilih Simpan perubahan.

Menggunakan AWS CLI

Contoh `put-object-retention` berikut menetapkan periode retensi pada objek *`my-image.fs`* dalam bucket berama *`example-s3-bucket1`* hingga 1 Januari 2025:

```
aws s3api put-object-retention --bucket example-s3-bucket1 --key my-image.fs --
retention='{ "Mode": "GOVERNANCE", "RetainUntilDate": "2025-01-01T00:00:00" }'
```

Untuk informasi dan contoh selengkapnya, lihat put-object-retention di Referensi Perintah AWS CLI .

> **Note**
>
> Anda dapat menjalankan AWS CLI perintah dari konsol dengan menggunakan AWS CloudShell. AWS CloudShell adalah shell pra-otentikasi berbasis browser yang dapat Anda luncurkan langsung dari file. AWS Management Console Untuk informasi lebih lanjut, lihat Apa itu CloudShell? dalam AWS CloudShell User Guide.

Penggunaan API REST

Anda dapat menggunakan API REST untuk menetapkan periode retensi pada objek. Untuk informasi selengkapnya, lihat PutObjectRetention di Referensi API Amazon Simple Storage Service.

Menggunakan AWS SDK

Untuk contoh cara mengatur periode retensi pada objek dengan AWS SDK, lihatGunakan `PutObjectRetention` dengan AWS SDK atau CLI.

Untuk contoh cara mendapatkan periode retensi pada objek dengan AWS SDK, lihatGunakan `GetObjectRetention` dengan AWS SDK atau CLI.

Untuk skenario interaktif yang mendemonstrasikan fitur Object Lock yang berbeda menggunakan AWS SDK, lihat. Bekerja dengan fitur kunci objek Amazon S3 menggunakan SDK AWS

Untuk informasi umum tentang penggunaan AWS SDK yang berbeda, lihatMengembangkan dengan Amazon S3 menggunakan SDK AWS.

Menetapkan atau memodifikasi periode retensi default pada bucket S3

Anda dapat menyetel atau mengubah periode retensi default pada bucket S3 menggunakan konsol Amazon S3, SDK AWS CLI AWS , atau Amazon S3 REST API. Anda menentukan durasi, baik dalam hitungan hari atau tahun, untuk lamanya perlindungan terhadap setiap versi objek yang ditempatkan di dalam bucket.

> **Important**
>
> - Jika Anda ingin menetapkan periode retensi default pada bucket, bucket harus sudah mengaktifkan Kunci Objek.
> - Jika Anda memiliki versi objek PUT yang memiliki mode dan periode retensi individual eksplisit dalam bucket, pengaturan Kunci Objek individual versi objek akan menimpa pengaturan retensi properti bucket apa pun.
> - Satu-satunya cara untuk menghapus objek di bawah mode kepatuhan sebelum tanggal retensi berakhir adalah dengan menghapus yang terkait Akun AWS.

Untuk informasi selengkapnya, lihat Periode retensi.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di daftar Bucket, pilih nama bucket yang ingin Anda tetapkan atau ubah periode retensi defaultnya.
4. Pilih tab Properti.
5. Di bawah Properti, gulir ke bawah ke bagian Kunci Objek, lalu pilih Edit.
6. Di bawah Retensi default, pilih Aktifkan untuk menetapkan retensi default atau Nonaktifkan untuk menghapus retensi default.
7. Jika Anda memilih Aktifkan, di bawah Mode retensi, pilih mode Tata Kelola atau mode Kepatuhan. Untuk informasi selengkapnya, lihat Mode retensi.
8. Di bawah Periode retensi default, pilih jumlah hari atau tahun yang Anda inginkan untuk periode retensi. Objek yang ditempatkan di bucket ini akan terkunci selama beberapa hari atau tahun. Untuk informasi selengkapnya, lihat Periode retensi.
9. Pilih Simpan perubahan.

Menggunakan AWS CLI

Contoh `put-object-lock-configuration` perintah berikut menetapkan periode retensi Kunci Objek 50 hari pada bucket yang bernama *`example-s3-bucket1`* menggunakan mode kepatuhan:

```
aws s3api put-object-lock-configuration --bucket example-s3-bucket1 --object-lock-
configuration='{ "ObjectLockEnabled": "Enabled", "Rule": { "DefaultRetention":
 { "Mode": "COMPLIANCE", "Days": 50 }}}'
```

Contoh `put-object-lock-configuration` berikut menghapus konfigurasi retensi default pada bucket:

```
aws s3api put-object-lock-configuration --bucket example-s3-bucket1 --object-lock-
configuration='{ "ObjectLockEnabled": "Enabled"}'
```

Untuk informasi dan contoh selengkapnya, lihat put-object-lock-configuration di Referensi Perintah AWS CLI .

> **Note**
>
> Anda dapat menjalankan AWS CLI perintah dari konsol dengan menggunakan AWS CloudShell. AWS CloudShell adalah shell pra-otentikasi berbasis browser yang dapat Anda luncurkan langsung dari file. AWS Management Console Untuk informasi lebih lanjut, lihat Apa itu CloudShell? dalam AWS CloudShell User Guide.

Penggunaan API REST

Anda dapat menggunakan REST API untuk menyetel periode retensi default pada bucket S3 yang ada. Untuk informasi selengkapnya, lihat PutObjectLockConfiguration dalam Referensi API Amazon Simple Storage Service.

Menggunakan AWS SDK

Untuk contoh cara menyetel periode retensi default pada bucket S3 yang ada dengan AWS SDK, lihat. Gunakan `PutObjectLockConfiguration` dengan AWS SDK atau CLI

Untuk skenario interaktif yang mendemonstrasikan fitur Object Lock yang berbeda menggunakan AWS SDK, lihat. Bekerja dengan fitur kunci objek Amazon S3 menggunakan SDK AWS

Untuk informasi umum tentang penggunaan AWS SDK yang berbeda, lihatMengembangkan dengan Amazon S3 menggunakan SDK AWS.

Menggunakan kelas penyimpanan Amazon S3

Setiap objek di Amazon S3 memiliki kelas penyimpanan yang terkait dengan objeknya. Misalnya, jika Anda mencantumkan objek di bucket S3, konsol akan menunjukkan kelas penyimpanan untuk semua objek di dalam daftar. Amazon S3 menawarkan serangkaian kelas penyimpanan untuk objek yang disimpan. Pilih kelas yang sesuai dengan skenario kasus penggunaan dan kebutuhan akses kinerja. Semua kelas penyimpanan ini menawarkan daya tahan tinggi.

Bagian berikut memberikan rincian dari berbagai kelas penyimpanan dan cara mengatur kelas penyimpanan untuk objek.

Topik

- Kelas penyimpanan untuk objek yang sering diakses

- Kelas penyimpanan untuk mengoptimalkan data secara otomatis dengan pola akses yang berubah atau tidak diketahui
- Kelas penyimpanan untuk objek yang jarang diakses
- Kelas penyimpanan untuk objek yang jarang diakses
- Kelas penyimpanan untuk Amazon S3 pada Outposts
- Membandingkan kelas penyimpanan Amazon S3
- Mengatur kelas penyimpanan objek

Kelas penyimpanan untuk objek yang sering diakses

Untuk kasus penggunaan yang sensitif terhadap kinerja (kasus yang memerlukan waktu akses milidetik) dan data yang sering diakses, Amazon S3 menyediakan kelas penyimpanan berikut:

- S3 Standard–Kelas penyimpanan default. Jika kelas penyimpanan tidak ditentukan saat mengunggah objek, Amazon S3 akan menetapkan kelas penyimpanan S3 Standard.
- S3 Express One Zone–Amazon S3 Express One Zone adalah kelas penyimpanan Amazon S3 tunggal berkinerja tinggi yang dibuat khusus untuk menghadirkan akses data milidetik satu digit yang konsisten untuk aplikasi Anda yang paling sensitif terhadap latensi. S3 Express One Zone adalah kelas penyimpanan objek cloud latensi terendah yang tersedia saat ini, dengan kecepatan akses data hingga 10x lebih cepat dan dengan biaya permintaan 50 persen lebih rendah dari Standar S3. Dengan S3 Express One Zone, data Anda disimpan secara berlebihan di beberapa perangkat dalam satu Zona Ketersediaan. Untuk informasi selengkapnya, lihat Apa itu S3 Express One Zone?.
- Reduced Redundancy–Kelas penyimpanan Reduced Redundancy Storage (RRS) dirancang untuk data non-kritis dan dapat direproduksi yang dapat disimpan dengan redundansi lebih sedikit dibandingkan kelas penyimpanan S3 Standard.

  > **Important**
  >
  > Sebaiknya hindari penggunaan kelas penyimpanan ini. Kelas penyimpanan S3 Standard lebih hemat biaya.

  Untuk ketahanan, objek RRS memiliki perkiraan kehilangan tahunan rata-rata sebesar 0,01 persen dari objek. Jika objek RRS hilang, ketika permintaan dibuat ke objek tersebut, Amazon S3 akan menampilkan kesalahan 405.

Kelas penyimpanan untuk mengoptimalkan data secara otomatis dengan pola akses yang berubah atau tidak diketahui

S3 Intelligent-Tiering adalah kelas penyimpanan Amazon S3 yang dirancang untuk mengoptimalkan biaya penyimpanan dengan secara otomatis memindahkan data ke tingkat akses yang paling hemat biaya, tanpa dampak kinerja atau overhead operasional. S3 Intelligent-Tiering adalah satu-satunya kelas penyimpanan cloud yang memberikan penghematan biaya otomatis dengan memindahkan data pada tingkat objek granular antar tingkat akses ketika pola akses berubah. S3 Intelligent-Tiering adalah kelas penyimpanan yang ideal ketika Anda ingin mengoptimalkan biaya penyimpanan untuk data yang pola aksesnya tidak diketahui atau berubah. Tidak ada biaya pengambilan untuk S3 Intelligent-Tiering.

Dengan sedikit biaya pemantauan objek dan otomatisasi bulanan, S3 Intelligent-Tiering memantau pola akses dan secara otomatis memindahkan objek yang belum diakses ke tingkat akses yang biayanya lebih rendah. S3 Intelligent-Tiering memberikan penghematan biaya penyimpanan otomatis dalam tiga tingkat akses latensi rendah dan throughput tinggi. Untuk data yang dapat diakses secara asinkron, Anda dapat memilih untuk mengaktifkan kemampuan pengarsipan otomatis di dalam kelas penyimpanan S3 Intelligent-Tiering. S3 Intelligent-Tiering dirancang untuk ketersediaan 99,9% dan daya tahan 99,99999%.

S3 Intelligent-Tiering secara otomatis menyimpan objek ke dalam tiga tingkatan akses:

- Frequent Access–Objek yang diunggah atau ditransisikan ke S3 Intelligent-Tiering secara otomatis akan disimpan di tingkat Frequent Access.
- Infrequent Access–S3 Intelligent-Tiering memindahkan objek yang belum diakses selama 30 hari berturut-turut ke tingkat Infrequent Access.
- Archive Instant Access–Dengan S3 Intelligent-Tiering, objek yang ada yang belum diakses selama 90 hari berturut-turut secara otomatis akan dipindahkan ke tingkat Archive Instant Access.

Selain tiga tingkatan ini, S3 Intelligent-Tiering menawarkan dua tingkatan akses arsip opsional:

- Archive Access–S3 Intelligent-Tiering memberikan opsi untuk mengaktifkan tingkat Archive Access untuk data yang dapat diakses secara asinkron. Setelah aktivasi, tingkat Archive Access secara otomatis akan mengarsipkan objek yang belum diakses selama minimal 90 hari berturut-turut.
- Deep Archive Access–S3 Intelligent-Tiering memberikan opsi untuk mengaktifkan tingkat Deep Archive Access untuk data yang dapat diakses secara asinkron. Setelah aktivasi, tingkat Deep

Archive Access secara otomatis akan mengarsipkan objek yang belum diakses selama minimal 180 hari berturut-turut.

> **Note**
>
> - Hanya aktifkan tingkat Archive Access selama 90 hari jika Anda ingin melewati tingkat Archive Instant Access. Tingkat Akses Arsip memberikan penyimpanan berbiaya sedikit lebih rendah dengan minute-to-hour waktu pengambilan. Tingkat Archive Instant Access memberikan akses milidetik dan kinerja dengan throughput tinggi.
> - Aktifkan tingkatan Archive Access dan Deep Archive Access hanya jika objek dapat diakses secara asinkron oleh aplikasi Anda. Jika objek yang Anda ambil disimpan di tingkat Archive Access atau Deep Archive Access, pulihkan objek terlebih dahulu dengan menggunakan `RestoreObject`.

Anda dapat memindahkan data yang baru dibuat ke S3 Intelligent-Tiering, mengaturnya sebagai kelas penyimpanan default. Anda juga dapat memilih untuk mengaktifkan salah satu atau kedua tingkatan akses arsip dengan menggunakan operasi `PutBucketIntelligentTieringConfiguration`API, konsol AWS CLI, atau Amazon S3. Untuk informasi selengkapnya tentang penggunaan S3 Intelligent-Tiering dan pengaktifan tingkatan akses arsip, lihat Menggunakan S3 Intelligent-Tiering.

Untuk mengakses objek di jenjang Archive Access atau Deep Archive Access, Anda harus mengembalikannya terlebih dahulu. Untuk informasi selengkapnya, lihat Memulihkan objek dari tingkat S3 Intelligent-Tiering Archive Access dan Deep Archive Access.

> **Note**
>
> Jika ukuran suatu objek kurang dari 128 KB, objek tersebut tidak dipantau dan tidak memenuhi syarat untuk penetapan tingkat otomatis. Objek yang lebih kecil akan selalu disimpan di tingkat Frequent Access. Untuk informasi selengkapnya tentang S3 Intelligent-Tiering, lihat Tingkat akses Tingkat Cerdas S3.

Kelas penyimpanan untuk objek yang jarang diakses

Kelas penyimpanan S3 Standard-IA dan S3 One Zone-IA dirancang untuk data yang berumur panjang dan jarang diakses. (IA singkatan dari infrequent access.) Objek S3 Standard-IA dan S3 One Zone-IA tersedia untuk akses milidetik (mirip dengan kelas penyimpanan S3 Standard). Amazon S3 membebankan biaya pengambilan untuk objek ini, sehingga objek tersebut paling cocok untuk data yang jarang diakses. Untuk informasi harga, lihat harga Amazon S3.

Misalnya, Anda bisa memilih kelas penyimpanan S3 Standard-IA dan S3 One Zone-IA untuk melakukan hal berikut:

- Untuk menyimpan cadangan.
- Untuk data lama yang tidak sering diakses, tetapi masih membutuhkan akses milidetik. Misalnya, saat mengunggah data, Anda dapat memilih kelas penyimpanan S3 Standard, dan menggunakan konfigurasi siklus hidup untuk memberi tahu Amazon S3 guna mentransisi objek ke kelas S3 Standard-IA atau S3 One Zone-IA.

  Untuk informasi lebih lanjut tentang manajemen siklus hidup, lihat Mengelola siklus hidup penyimpanan Anda.

> **Note**
>
> Kelas penyimpanan S3 Standard-IA dan S3 One Zone-IA cocok untuk objek yang lebih besar dari 128 KB untuk disimpan setidaknya selama 30 hari. Jika suatu objek berukuran kurang dari 128 KB, Amazon S3 akan mengenakan biaya sebesar 128 KB. Jika menghapus objek sebelum akhir periode durasi penyimpanan minimum 30 hari, Anda akan dikenakan biaya selama 30 hari. Objek yang dihapus, menimpa, atau dialihkan ke kelas penyimpanan lain sebelum 30 hari akan dikenakan biaya penggunaan penyimpanan normal ditambah biaya prorata untuk sisa minimum 30 hari. Untuk informasi harga, lihat harga Amazon S3.

Kelas penyimpanan ini memiliki perbedaan sebagai berikut:

- S3 Standard-IA–Amazon S3 menyimpan data objek secara berlebihan di beberapa Zona Ketersediaan yang terpisah secara geografis (mirip dengan kelas penyimpanan S3 Standard). Objek S3 Standard-IA tahan terhadap hilangnya Zona Ketersediaan. Kelas penyimpanan ini menawarkan ketersediaan dan ketahanan yang lebih besar dibandingkan kelas S3 One Zone-IA.

- S3 One Zone-IA–Amazon S3 menyimpan data objek hanya dalam satu Zona Ketersediaan, yang membuatnya lebih murah daripada S3 Standard-IA. Namun, data tersebut tidak tahan terhadap kehilangan fisik Zona Ketersediaan yang diakibatkan oleh bencana, seperti gempa bumi dan banjir. Kelas penyimpanan S3 One Zone-IA sama tahan lamanya dengan S3 Standard-IA, tapi ketersediaannya lebih sedikit dan ketahanannya kurang. Untuk perbandingan ketahanan dan ketersediaan kelas penyimpanan, lihat Membandingkan kelas penyimpanan Amazon S3 di akhir bagian ini. Untuk informasi harga, lihat harga Amazon S3.

Sebaiknya lakukan hal berikut:

- S3 Standard-IA–Gunakan untuk salinan data utama atau satu-satunya yang tidak dapat dibuat ulang.
- S3 One Zone-IA–Gunakan jika Anda dapat membuat ulang data jika Zona Ketersediaan gagal, dan untuk replika objek saat mengonfigurasi S3 Cross-Region Replication (CRR).

Kelas penyimpanan untuk objek yang jarang diakses

Kelas S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval, dan S3 Glacier Deep Archive dirancang untuk penyimpanan data jangka panjang dan pengarsipan data berbiaya rendah. Kelas penyimpanan ini menawarkan daya tahan dan ketahanan yang sama dengan kelas penyimpanan S3 Standard dan S3 Standard-IA. Untuk informasi lebih lanjut tentang kelas penyimpanan S3 Glacier, lihat. Penyimpanan data jangka panjang menggunakan kelas penyimpanan S3 Glacier

Amazon S3 menyediakan kelas penyimpanan S3 Glacier berikut:

- S3 Glacier Instant Retrieval — Gunakan untuk data jangka panjang yang jarang diakses dan membutuhkan pengambilan milidetik. Data dalam kelas penyimpanan ini tersedia untuk akses real-time.
- S3 Glacier Flexer–Gunakan untuk arsip yang sebagian datanya mungkin perlu diambil dalam hitungan menit. Data dalam kelas penyimpanan ini diarsipkan, dan tidak tersedia untuk akses real-time.
- S3 Glacier Deep Archive–Gunakan untuk pengarsipan data yang jarang diakses. Data dalam kelas penyimpanan ini diarsipkan, dan tidak tersedia untuk akses real-time.

Mengambil objek yang diarsipkan

Anda dapat mengatur kelas penyimpanan suatu objek ke S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive dengan cara yang sama seperti yang Anda lakukan untuk kelas penyimpanan lainnya seperti yang dijelaskan di bagian Mengatur kelas penyimpanan objek. Namun, objek S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive diarsipkan, dan tidak tersedia untuk akses real-time. Untuk informasi selengkapnya, lihat Penyimpanan arsip.

> **Note**
>
> Saat Anda menggunakan kelas penyimpanan S3 Glacier, objek Anda tetap berada di Amazon S3. Anda tidak dapat mengaksesnya secara langsung melalui layanan Amazon S3 Glacier yang terpisah. Untuk informasi tentang layanan Amazon S3 Glacier, lihat Panduan Pengembang Amazon S3 Glacier.

Kelas penyimpanan untuk Amazon S3 pada Outposts

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di AWS Outposts sumber daya serta menyimpan serta mengambil objek lokal untuk aplikasi yang memerlukan akses data lokal, pemrosesan data lokal, dan residensi data. Anda dapat menggunakan operasi dan fitur API yang sama AWS Outposts seperti yang Anda lakukan di Amazon S3, termasuk kebijakan akses, enkripsi, dan penandaan. Anda dapat menggunakan S3 di Outposts melalui AWS Management Console,, SDK AWS CLI AWS , atau REST API.

S3 di Outposts menyediakan kelas penyimpanan baru, S3 Outposts (`OUTPOSTS`). Kelas penyimpanan S3 Outposts hanya tersedia untuk objek yang disimpan di bucket yang ada di Outposts. Jika Anda mencoba menggunakan kelas penyimpanan ini dengan bucket S3 di Wilayah AWS, `InvalidStorageClass` kesalahan terjadi. Selain itu, jika Anda mencoba menggunakan kelas penyimpanan S3 lain dengan objek yang disimpan di S3 pada bucket Outposts, kesalahan yang sama akan terjadi.

Objek yang disimpan di kelas penyimpanan S3 Outposts (`OUTPOSTS`) selalu dienkripsi menggunakan enkripsi di sisi server dengan kunci enkripsi terkelola Amazon S3 (SSE-S3). Untuk informasi selengkapnya, lihat Menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3).

Anda juga dapat secara eksplisit memilih untuk mengenkripsi objek yang disimpan di kelas penyimpanan S3 Outposts dengan menggunakan enkripsi di sisi server dengan kunci enkripsi yang

disediakan pelanggan (SSE-C). Untuk informasi selengkapnya, lihat Menggunakan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C).

> **Note**
>
> S3 di Outposts tidak mendukung enkripsi sisi server AWS Key Management Service dengan kunci AWS KMS() (SSE-KMS).

Untuk informasi lebih lanjut tentang S3 di Outposts. lihat Apa itu Amazon S3 di Outposts?

Membandingkan kelas penyimpanan Amazon S3

Tabel berikut membandingkan kelas penyimpanan, termasuk ketersediaan, daya tahan, durasi penyimpanan minimum, dan pertimbangan lainnya.

Storage Class	Designed for	Durability (designed for)	Availability (designed for)	Availability Zones	Min storage duration	Min billable object size	Other Considerations
STANDARD	Frequently accessed data	99.999999999%	99.99%	>= 3	None	None	None
STANDARD_IA	Long-lived, infrequently accessed data	99.999999999%	99.9%	>= 3	30 days	128 KB	Per GB retrieval fees apply.
INTELLIGENT_TIERING	Long-lived data with changing or unknown access patterns	99.999999999%	99.9%	>= 3	30 days	None	Monitoring and automation fees per object apply. No retrieval fees.
ONEZONE_IA	Long-lived, infrequently accessed, non-critical data	99.999999999%	99.5%	1	30 days	128 KB	Per GB retrieval fees apply. Not resilient to the loss of the Availability Zone.
GLACIER	Long-term data archiving with retrieval times ranging from minutes to hours	99.999999999%	99.99% (after you restore objects)	>= 3	90 days	None	Per GB retrieval fees apply. You must first restore archived objects before you can access them. For more information, see Restoring Archived Objects.
DEEP_ARCHIVE	Archiving rarely accessed data with a default retrieval time of 12 hours	99.999999999%	99.99% (after you restore objects)	>= 3	180 days	None	Per GB retrieval fees apply. You must first restore archived objects before you can access them. For more information, see Restoring Archived Objects.
RRS (Not recommended)	Frequently accessed, non-critical data	99.99%	99.99%	>= 3	None	None	None

* S3 Glacier Flexible Retrieval membutuhkan 40 KB metadata tambahan untuk setiap objek yang diarsipkan. Hal ini termasuk metadata sebesar 32 KB yang dikenakan tarif pada tingkat S3 Glacier Flexible Retrieval (diperlukan untuk mengidentifikasi dan mengambil data), dan data tambahan sebesar 8 KB yang dikenakan tarif S3 Standard. Tarif S3 Standard diperlukan untuk mempertahankan nama dan metadata yang ditentukan pengguna untuk objek yang diarsipkan ke S3 Glacier Flexible Retrieval. Untuk informasi selengkapnya tentang kelas penyimpanan, lihat Kelas penyimpanan Amazon S3.

** S3 Glacier Deep Archive membutuhkan 40 KB metadata tambahan untuk setiap objek yang diarsipkan. Hal ini termasuk metadata sebesar 32 KB yang dikenakan tarif S3 Glacier Deep Archive (diperlukan untuk mengidentifikasi dan mengambil data), dan data tambahan sebesar 8 KB yang dikenakan tarif S3 Standard. Tarif S3 Standard diperlukan untuk mempertahankan nama dan metadata yang ditentukan pengguna untuk objek yang diarsipkan ke Amazon S3 Glacier Deep Archive. Untuk informasi selengkapnya tentang kelas penyimpanan, lihat Kelas penyimpanan Amazon S3.

Perlu diketahui bahwa semua kelas penyimpanan kecuali S3 One Zone-IA dan S3 Express One Zone dirancang agar tahan terhadap kehilangan fisik Zona Ketersediaan akibat bencana. Pertimbangkan juga biaya, selain kebutuhan kinerja dari skenario aplikasi. Untuk harga kelas penyimpanan, lihat harga Amazon S3.

Mengatur kelas penyimpanan objek

Untuk mengatur dan memperbarui kelas penyimpanan objek, Anda dapat menggunakan konsol Amazon S3, AWS SDK, atau (). AWS Command Line Interface AWS CLI Semua pendekatan ini menggunakan operasi API Amazon S3 untuk mengirim permintaan ke Amazon S3.

Operasi API Amazon S3 mendukung pengaturan (atau pembaruan) kelas penyimpanan objek sebagai berikut:

- Ketika menciptakan objek baru, Anda dapat menentukan kelas penyimpanannya. Misalnya, saat membuat objek menggunakan operasi API PUT Objek, POST Objek, dan Memulai Unggahan Multibagian, Anda dapat menambahkan header permintaan `x-amz-storage-class` untuk menentukan kelas penyimpanan. Jika header ini tidak ditambahkan, Amazon S3 akan menggunakan S3 Standard, kelas penyimpanan default.
- Anda juga dapat mengubah kelas penyimpanan objek yang sudah disimpan di Amazon S3 ke kelas penyimpanan lain dengan membuat salinan objek menggunakan operasi API Objek PUT-Salin. Tetapi, Anda tidak dapat menggunakan Objek PUT-Salin untuk menyalin objek yang disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive. Anda juga tidak dapat bertransisi dari S3 One Zone-IA ke S3 Glacier Instant Retrieval.

  Anda harus menyalin objek di dalam bucket yang sama dengan menggunakan nama kunci yang sama dan menentukan header permintaan sebagai berikut:

  - Atur header `x-amz-metadata-directive` ke `COPY`.
  - Atur header `x-amz-storage-class` ke kelas penyimpanan yang ingin digunakan.

Di dalam bucket yang mengaktifkan Penentuan Versi, Anda tidak dapat mengubah kelas penyimpanan versi objek tertentu. Saat objek disalin, Amazon S3 akan memberikan ID versi yang baru.

- Anda dapat mengubah kelas penyimpanan objek menggunakan konsol Amazon S3 jika ukuran objek kurang dari 160 GB. Jika ukurannya lebih besar, sebaiknya tambahkan konfigurasi S3 Lifecycle untuk mengubah kelas penyimpanan objek.
- Jika Anda menggunakan konsol Amazon S3 untuk mengubah kelas penyimpanan untuk objek yang memiliki tag yang ditentukan pengguna, Anda harus memiliki izin. `s3:GetObjectTagging` Jika Anda mengubah kelas penyimpanan untuk objek yang tidak memiliki tag yang ditentukan pengguna tetapi berukuran lebih dari 16 MB, Anda juga harus memiliki izin. `s3:GetObjectTagging` Jika kebijakan bucket tujuan menolak `s3:GetObjectTagging` tindakan, kelas penyimpanan untuk objek akan diperbarui, tetapi tag yang ditentukan pengguna akan dihapus dari objek, dan Anda akan menerima kesalahan.
- Anda dapat mengarahkan Amazon S3 untuk mengubah kelas penyimpanan objek dengan menambahkan konfigurasi S3 Lifecycle ke bucket. Untuk informasi selengkapnya, lihat Mengelola siklus hidup penyimpanan Anda.
- Saat menyetel konfigurasi replikasi, Anda dapat mengatur kelas penyimpanan untuk objek yang direplikasi ke kelas penyimpanan lain. Tetapi, Anda tidak dapat mereplikasi objek yang disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive. Untuk informasi selengkapnya, lihat Konfigurasi Replikasi.

Membatasi izin kebijakan akses untuk kelas penyimpanan tertentu

Saat memberikan izin kebijakan akses untuk operasi Amazon S3, Anda dapat menggunakan kunci syarat `s3:x-amz-storage-class` untuk membatasi kelas penyimpanan mana yang akan digunakan saat menyimpan objek yang diunggah. Misalnya, saat Anda memberikan izin `s3:PutObject`, Anda dapat membatasi pengunggahan objek ke kelas penyimpanan tertentu. Untuk contoh kebijakan, lihat Contoh: Membatasi unggahan objek ke objek dengan kelas penyimpanan tertentu.

Untuk informasi selengkapnya tentang ketentuan penggunaan di dalam kebijakan dan daftar lengkap kunci syarat Amazon S3, lihat topik berikut:

- Kunci tindakan, sumber daya, dan kondisi untuk Amazon S3 di Referensi Otorisasi Layanan
- Contoh kebijakan bucket menggunakan tombol kondisi

Penyimpanan data jangka panjang menggunakan kelas penyimpanan S3 Glacier

Amazon S3 menawarkan beberapa kelas penyimpanan S3 Glacier yang dirancang untuk memberikan solusi hemat biaya untuk menyimpan data jangka panjang yang tidak sering diakses. Kelas penyimpanan S3 Glacier adalah:

- S3 Glacier Instant Retrieval
- S3 Glacier Flexible Retrieval
- S3 Glacier Deep Archive

Anda memilih salah satu kelas penyimpanan ini berdasarkan seberapa sering Anda mengakses data Anda dan seberapa cepat Anda perlu mengambilnya. Masing-masing kelas penyimpanan ini menawarkan daya tahan dan ketahanan yang sama dengan kelas penyimpanan Standar S3, tetapi dengan biaya penyimpanan yang lebih rendah. Untuk informasi lebih lanjut tentang kelas penyimpanan S3 Glacier, lihat https://aws.amazon.com/s3/storage-classes/glacier/.

Topik

- Membandingkan kelas penyimpanan S3 Glacier
- S3 Glacier Instant Retrieval
- S3 Glacier Flexible Retrieval
- S3 Glacier Deep Archive
- Penyimpanan arsip
- Bagaimana kelas penyimpanan ini berbeda dari layanan S3 Glacier

Membandingkan kelas penyimpanan S3 Glacier

Setiap kelas penyimpanan S3 Glacier memiliki durasi penyimpanan minimum untuk semua objek. Jika Anda menghapus, menimpa, atau mentransisikan objek ke kelas penyimpanan yang berbeda sebelum minimum, Anda akan dikenakan biaya untuk durasi penyimpanan minimum penuh.

Beberapa kelas penyimpanan S3 Glacier adalah arsip, yang berarti objek yang disimpan di kelas tersebut diarsipkan dan tidak tersedia untuk akses real-time. Untuk informasi selengkapnya, lihat Penyimpanan arsip.

Kelas penyimpanan yang dirancang untuk pola akses yang lebih jarang dengan waktu pengambilan yang lebih lama menawarkan biaya penyimpanan yang lebih rendah. Untuk informasi harga, lihathttps://aws.amazon.com/s3/pricing/.

Tabel berikut merangkum poin-poin penting yang perlu dipertimbangkan ketika memilih kelas penyimpanan S3 Glacier:

S3 Glacier Instant Retrieval

Kami merekomendasikan penggunaan S3 Glacier Instant Retrieval untuk data jangka panjang yang diakses sekali per kuartal dan membutuhkan waktu pengambilan milidetik. Kelas penyimpanan ini sangat ideal untuk kasus penggunaan yang sensitif terhadap kinerja seperti hosting gambar, aplikasi berbagi file, dan menyimpan catatan medis untuk akses selama janji temu.

Kelas penyimpanan S3 Glacier Instant Retrieval menawarkan akses real-time ke objek Anda dengan kinerja latensi dan throughput yang sama dengan kelas penyimpanan IA Standar S3. Jika dibandingkan dengan S3 Standard-IA, S3 Glacier Instant Retrieval memiliki biaya penyimpanan yang lebih rendah tetapi biaya akses data yang lebih tinggi.

Ada ukuran objek minimum 128 KB untuk data yang disimpan di kelas penyimpanan S3 Glacier Instant Retrieval. Kelas penyimpanan ini juga memiliki jangka waktu penyimpanan minimum 90 hari.

S3 Glacier Flexible Retrieval

Kami merekomendasikan penggunaan S3 Glacier Flexible Retrieval untuk data arsip yang diakses satu hingga dua kali setahun dan tidak memerlukan akses langsung. S3 Glacier Flexible Retrieval menawarkan waktu pengambilan yang fleksibel untuk membantu Anda menyeimbangkan biaya, dengan waktu akses mulai dari beberapa menit hingga jam, dan pengambilan massal gratis. Kelas penyimpanan ini sangat ideal untuk cadangan dan pemulihan bencana.

Objek yang disimpan di S3 Glacier Flexible Retrieval diarsipkan dan tidak tersedia untuk akses real-time. Untuk informasi selengkapnya, lihat Penyimpanan arsip. Untuk mengakses objek ini, pertama-tama Anda memulai permintaan pemulihan yang membuat salinan sementara objek yang dapat Anda akses saat permintaan selesai. Untuk informasi, lihat Bekerja dengan objek yang diarsipkan. Saat memulihkan objek, Anda dapat memilih tingkat pengambilan untuk memenuhi kasus penggunaan Anda, dengan biaya lebih rendah untuk waktu pemulihan yang lebih lama.

Tingkatan pengambilan berikut tersedia untuk S3 Glacier Flexible Retrieval:

- Pengambilan dipercepat — Biasanya mengembalikan objek dalam 1—5 menit. Pengambilan yang dipercepat tunduk pada permintaan, jadi untuk memastikan Anda memiliki waktu pemulihan yang andal dan dapat diprediksi, kami sarankan Anda membeli kapasitas pengambilan yang disediakan. Untuk informasi selengkapnya, lihat Kapasitas yang disediakan.
- Pengambilan standar — Biasanya mengembalikan objek dalam 3-5 jam, atau dalam 1 menit hingga 5 jam saat Anda menggunakan Operasi Batch S3. Untuk informasi selengkapnya, lihat Memulihkan objek dengan Operasi Batch.
- Pengambilan massal — Biasanya mengembalikan objek dalam waktu 5-12 jam. Pengambilan massal gratis.

Durasi penyimpanan minimum untuk objek di kelas penyimpanan S3 Glacier Flexible Retrieval adalah 90 hari.

Pengambilan Fleksibel Gletser S3 membutuhkan 40 KB metadata tambahan untuk setiap objek. Ini termasuk 32 KB metadata yang diperlukan untuk mengidentifikasi dan mengambil data Anda, yang dibebankan pada tingkat default untuk S3 Glacier Flexible Retrieval. Data 8 KB tambahan diperlukan untuk mempertahankan nama dan metadata yang ditentukan pengguna untuk objek yang diarsipkan, dan dibebankan pada tarif Standar S3.

S3 Glacier Deep Archive

Sebaiknya gunakan S3 Glacier Deep Archive untuk arsip data yang diakses kurang dari setahun sekali. Kelas penyimpanan ini dirancang untuk mempertahankan kumpulan data selama beberapa tahun untuk memenuhi persyaratan kepatuhan dan juga dapat digunakan untuk cadangan atau pemulihan bencana atau data yang jarang diakses yang dapat Anda tunggu hingga 72 jam untuk diambil. S3 Glacier Deep Archive adalah opsi penyimpanan dengan biaya terendah di AWS.

Objek yang disimpan di S3 Glacier Deep Archive diarsipkan dan tidak tersedia untuk akses real-time. Untuk informasi selengkapnya, lihat Penyimpanan arsip. Untuk mengakses objek ini, pertama-tama Anda memulai permintaan pemulihan yang membuat salinan sementara objek yang dapat Anda akses saat permintaan selesai. Untuk informasi, lihat Bekerja dengan objek yang diarsipkan. Saat memulihkan objek, Anda dapat memilih tingkat pengambilan untuk memenuhi kasus penggunaan Anda, dengan biaya lebih rendah untuk waktu pemulihan yang lebih lama.

Tingkatan pengambilan berikut tersedia untuk S3 Glacier Deep Archive:

- Pengambilan standar — Biasanya mengembalikan objek dalam waktu 12 jam, atau dalam waktu 9-12 jam saat Anda menggunakan Operasi Batch S3. Untuk informasi selengkapnya, lihat Memulihkan objek dengan Operasi Batch.
- Pengambilan massal — Biasanya mengembalikan objek dalam waktu 48 jam dengan sebagian kecil dari biaya tingkat pengambilan Standar.

Durasi penyimpanan minimum untuk objek di kelas penyimpanan S3 Glacier Deep Archive adalah 180 hari.

S3 Glacier Deep Archive membutuhkan 40 KB metadata tambahan untuk setiap objek. Ini termasuk 32 KB metadata yang diperlukan untuk mengidentifikasi dan mengambil data Anda, yang dibebankan pada tarif default untuk S3 Glacier Deep Archive. Data 8 KB tambahan diperlukan untuk mempertahankan nama dan metadata yang ditentukan pengguna untuk objek yang diarsipkan, dan dibebankan pada tarif Standar S3.

Penyimpanan arsip

S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive adalah kelas penyimpanan arsip. Ini berarti bahwa ketika Anda menyimpan objek di kelas penyimpanan ini objek tersebut diarsipkan, dan tidak dapat diakses secara langsung. Untuk mengakses objek yang diarsipkan, Anda mengirimkan permintaan pemulihan untuk itu, dan kemudian menunggu layanan memulihkan objek. Permintaan pemulihan mengembalikan salinan sementara objek, dan salinan itu dihapus ketika durasi yang Anda tentukan dalam permintaan berakhir. Untuk mengetahui informasi selengkapnya, lihat Bekerja dengan objek yang diarsipkan.

Kelas penyimpanan ini membutuhkan 40 KB metadata tambahan untuk setiap objek yang diarsipkan. Ini termasuk 32 KB metadata yang diperlukan untuk mengidentifikasi dan mengambil data Anda, yang dibebankan pada tingkat default untuk kelas penyimpanan tersebut. Data 8 KB tambahan diperlukan untuk mempertahankan nama dan metadata yang ditentukan pengguna untuk objek yang diarsipkan, dan dibebankan pada tarif Standar S3.

Objek di kelas penyimpanan ini ditagih dengan tarif kelas penyimpanan Standar S3 saat Anda mengunggahnya menggunakan unggahan multibagian. Untuk informasi selengkapnya, lihat Unggahan multibagian dan harga.

Anda dapat memulihkan objek yang diarsipkan di kelas penyimpanan ini dengan hingga 1.000 transaksi per detik (TPS) permintaan pemulihan objek per akun per akun. Wilayah AWS

Bagaimana kelas penyimpanan ini berbeda dari layanan S3 Glacier

Kelas penyimpanan S3 Glacier adalah bagian dari layanan Amazon S3 dan menyimpan data sebagai objek dalam ember S3. Anda dapat mengelola objek di kelas penyimpanan ini menggunakan konsol S3 atau secara terprogram menggunakan API atau SDK S3. Saat menyimpan objek di kelas penyimpanan S3 Glacier, Anda dapat menggunakan fitur S3 seperti enkripsi lanjutan, penandaan objek, dan konfigurasi Siklus Hidup S3 untuk membantu mengelola aksesibilitas dan biaya data.

> **Important**
>
> Sebaiknya gunakan kelas penyimpanan S3 Glacier dalam layanan Amazon S3 untuk semua data jangka panjang Anda.

Layanan Amazon S3 Glacier (S3 Glacier) adalah layanan terpisah yang menyimpan data sebagai arsip di dalam brankas. Layanan ini tidak mendukung fitur Amazon S3 dan tidak menyediakan dukungan konsol untuk operasi pengunggahan dan pengunduhan data. Kami tidak menyarankan menggunakan layanan S3 Glacier untuk data jangka panjang Anda. Data yang disimpan dalam layanan ini tidak dapat diakses dari layanan Amazon S3. Jika Anda mencari informasi tentang layanan S3 Glacier, lihat Panduan Pengembang Amazon S3 Glacier. Untuk mentransfer data dari layanan Amazon S3 Glacier ke kelas penyimpanan di Amazon S3 lihat Transfer Data dari Amazon S3 Glacier Vaults ke Amazon S3 di perpustakaan solusi. AWS

Amazon S3 Intelligent-Tiering

Kelas penyimpanan S3 Intelligent-Tiering dirancang untuk mengoptimalkan biaya penyimpanan dengan secara otomatis memindahkan data ke jenjang akses yang paling hemat biaya jika pola akses berubah, tanpa biaya operasional atau berdampak pada kinerja. Untuk pemantauan objek bulanan dan biaya otomatisasi, S3 Intelligent-Tiering memantau pola akses dan secara otomatis memindahkan objek yang belum diakses ke jenjang akses berbiaya lebih rendah.

S3 Intelligent-Tiering memberikan penghematan biaya penyimpanan otomatis dalam tiga tingkat akses latensi rendah dan throughput tinggi. Untuk data yang dapat diakses secara asinkron, Anda dapat memilih untuk mengaktifkan kemampuan pengarsipan otomatis dalam kelas penyimpanan S3 Intelligent-Tiering. Tidak ada biaya pengambilan dalam S3 Intelligent-Tiering. Jika objek di tingkat Akses Jarang atau Tingkat Akses Instan Arsip diakses nanti, maka secara otomatis dipindahkan kembali ke tingkat Akses Sering. Tidak ada biaya jenjang tambahan jika objek dipindahkan di antara jenjang akses dalam kelas penyimpanan S3 Intelligent-Tiering.

S3 Intelligent-Tiering adalah kelas penyimpanan yang direkomendasikan untuk data dengan pola akses yang tidak diketahui, berubah, atau tidak dapat diprediksi, terlepas dari ukuran objek atau periode retensi, seperti data lake, analisis data, dan aplikasi baru.

Untuk informasi tentang menggunakan S3 Intelligent-Tiering, lihat bagian berikut:

Topik

- Cara kerja S3 Intelligent-Tiering
- Menggunakan S3 Intelligent-Tiering
- Mengelola S3 Intelligent-Tiering

Cara kerja S3 Intelligent-Tiering

Kelas penyimpanan Amazon S3 Intelligent-Tiering secara otomatis menyimpan objek dalam tiga tingkatan akses. Satu tingkat dioptimalkan untuk akses yang sering, satu tingkat biaya lebih rendah dioptimalkan untuk akses yang jarang, dan tingkat biaya sangat rendah lainnya dioptimalkan untuk data yang jarang diakses. Untuk biaya pemantauan dan otomatisasi objek bulanan yang rendah, S3 Intelligent-Tiering memantau pola akses dan secara otomatis memindahkan objek ke tingkat Akses Jarang ketika belum diakses selama 30 hari berturut-turut. Setelah 90 hari tanpa akses, objek dipindahkan ke tingkat Akses Instan Arsip tanpa dampak kinerja atau overhead operasional.

Untuk mendapatkan biaya penyimpanan terendah untuk data yang dapat diakses dalam hitungan menit hingga jam, aktifkan kemampuan pengarsipan untuk menambahkan dua tingkatan akses tambahan. Anda dapat meratakan objek ke tingkat Akses Arsip, tingkat Akses Arsip Dalam, atau keduanya. Dengan Akses Arsip, S3 Intelligent-Tiering memindahkan objek yang belum diakses selama minimal 90 hari berturut-turut ke tingkat Akses Arsip. Dengan Deep Archive Access, S3 Intelligent-Tiering memindahkan objek ke tingkat Deep Archive Access setelah minimal 180 hari berturut-turut tanpa akses. Untuk kedua tingkatan, Anda dapat mengonfigurasi jumlah hari tanpa akses berdasarkan kebutuhan Anda.

Tindakan berikut merupakan akses yang mencegah tingkatan objek Anda ke tingkat Akses Arsip atau tingkat Akses Arsip Dalam:

- Mengunduh atau menyalin objek melalui konsol Amazon S3.
- MemohonCopyObject,UploadPartCopy, atau mereplikasi objek dengan S3 Batch Replication. Dalam kasus ini, objek sumber dari operasi salinan atau replikasi berjenjang.

- MemohonGetObject,PutObject,RestoreObject,CompleteMultipartUpload,ListParts, atauSelectObjectContent.

Misalnya, jika objek Anda diakses melalui`SelectObjectContent`sebelum jumlah hari yang Anda tentukan tanpa akses (misalnya, 180 hari), tindakan itu mengatur ulang timer. Objek Anda tidak akan pindah ke tingkat Akses Arsip atau tingkat Akses Arsip Dalam hingga waktu setelah yang terakhir`SelectObjectContent`permintaan mencapai jumlah hari yang Anda tentukan.

Jika objek di tingkat Akses Jarang atau tingkat Akses Instan Arsip diakses nanti, objek tersebut secara otomatis dipindahkan kembali ke tingkat Akses Sering.

Tindakan berikut merupakan akses yang secara otomatis memindahkan objek dari tingkat Akses Jarang atau tingkat Akses Instan Arsip kembali ke tingkat Akses Sering:

- Mengunduh atau menyalin objek melalui konsol Amazon S3.
- MemohonCopyObject,UploadPartCopy, atau mereplikasi objek dengan Batch Replication. Dalam kasus ini, objek sumber dari operasi salinan atau replikasi berjenjang.
- MemohonGetObject,PutObject,RestoreObject,CompleteMultipartUpload, atauListParts.

Tindakan lainnyajanganmerupakan akses yang secara otomatis memindahkan objek dari tingkat Akses Jarang atau tingkat Akses Instan Arsip kembali ke tingkat Akses Sering. Berikut ini adalah contoh, bukan daftar definitif, dari tindakan tersebut:

- MemohonHeadObject,GetObjectTagging,PutObjectTagging,ListObjects,ListObjectsV2, atauListObjectVersions.
- MemohonSelectObjectContentbukan merupakan akses yang meningkatan objek hingga tingkat Akses Sering. Selain itu, ini tidak mencegah tiering objek turun dari tingkat Akses Sering ke tingkat Akses Jarang, dan kemudian ke tingkat Akses Instan Arsip.

Anda dapat mengonfigurasi S3 Intelligent-Tiering sebagai kelas penyimpanan default untuk data yang baru dibuat dengan menentukan`INTELLIGENT-TIERING`dalam dirimuPutBucketIntelligentTieringConfigurationpermintaan header. S3 Intelligent-Tiering dirancang untuk ketersediaan 99,9% dan daya tahan 99,999999999%.

> **Note**
>
> Jika ukuran objek kurang dari 128 KB, itu tidak dipantau dan tidak memenuhi syarat untuk tiering otomatis. Objek yang lebih kecil selalu disimpan di tingkat Frequent Access.

Tingkat akses Tingkat Cerdas S3

Bagian berikut menjelaskan berbagai tingkatan akses otomatis dan opsional. Ketika objek bergerak di antara tingkatan akses, kelas penyimpanan tetap sama (S3 Intelligent-Tiering).

Tingkat Akses Sering (otomatis)

Ini adalah tingkat akses default tempat objek apa pun yang dibuat atau dialihkan ke S3 Intelligent-Tiering memulai siklus hidupnya. Sebuah objek tetap berada di tingkat ini selama sedang diakses. Tingkat Frequent Access memberikan latensi rendah dan kinerja throughput tinggi.

Tingkat Akses Jarang (otomatis)

Jika objek tidak diakses selama 30 hari berturut-turut, objek bergerak ke tingkat Akses Jarang. Tingkat Akses Jarang memberikan latensi rendah dan kinerja throughput tinggi.

Tingkat Akses Instan Arsip (otomatis)

Jika objek tidak diakses selama 90 hari berturut-turut, objek akan berpindah ke tingkat Akses Instan Arsip. Tingkat Akses Instan Arsip memberikan latensi rendah dan kinerja throughput tinggi.

Tingkat Akses Arsip (opsional)

S3 Intelligent-Tiering memberi Anda opsi untuk mengaktifkan tingkat Akses Arsip untuk data yang dapat diakses secara asinkron. Setelah aktivasi, tingkat Akses Arsip secara otomatis mengarsipkan objek yang belum diakses selama minimal 90 hari berturut-turut. Anda dapat memperpanjang waktu akses terakhir untuk pengarsipan hingga maksimum 730 hari. Tingkat Akses Arsip memiliki kinerja yang sama dengan Pengambilan Fleksibel Gletser S3 kelas penyimpanan.

Waktu pengambilan standar untuk tingkat akses ini dapat berkisar antara 3-5 jam. Jika Anda memulai permintaan pemulihan dengan menggunakan Operasi Batch S3, pemulihan Anda dimulai dalam beberapa menit. Untuk informasi selengkapnya tentang opsi dan waktu pengambilan, lihat the section called “Memulihkan objek dari tingkat S3 Intelligent-Tiering Archive Access dan Deep Archive Access”.

> **Note**
>
> Hanya aktifkan tingkat Akses Arsip selama 90 hari jika Anda ingin melewati tingkat Akses Instan Arsip. Tingkat Akses Arsip memberikan biaya penyimpanan yang sedikit lebih rendah, denganminute-to-hourwaktu pengambilan. Tingkat Akses Instan Arsip memberikan akses milidetik dan kinerja throughput tinggi.

Tingkat Akses Arsip Dalam (opsional)

S3 Intelligent-Tiering memberi Anda opsi untuk mengaktifkan tingkat Akses Arsip Dalam untuk data yang dapat diakses secara asinkron. Setelah aktivasi, tingkat Akses Arsip Dalam secara otomatis mengarsipkan objek yang belum diakses selama minimal 180 hari berturut-turut. Anda dapat memperpanjang waktu akses terakhir untuk pengarsipan hingga maksimum 730 hari. Tingkat Akses Arsip Dalam memiliki kinerja yang sama denganArsip Dalam Gletser S3kelas penyimpanan.

Pengambilan standar objek dalam tingkat akses ini terjadi dalam waktu 12 jam. Jika Anda memulai permintaan pemulihan dengan menggunakan Operasi Batch S3, pemulihan dimulai dalam waktu 9 jam. Untuk informasi selengkapnya tentang opsi dan waktu pengambilan, lihatthe section called “Memulihkan objek dari tingkat S3 Intelligent-Tiering Archive Access dan Deep Archive Access”.

> **Note**
>
> Aktifkan tingkatan Akses Arsip dan Akses Arsip Dalam hanya jika objek Anda dapat diakses secara asinkron oleh aplikasi Anda. Jika objek yang Anda ambil disimpan di tingkat Akses Arsip atau Akses Arsip Dalam, Anda harus terlebih dahulu memulihkan objek dengan menggunakan`RestoreObject`operasi.

Menggunakan S3 Intelligent-Tiering

Anda dapat menggunakan kelas penyimpanan S3 Intelligent-Tiering untuk mengoptimalkan biaya penyimpanan secara otomatis. S3 Intelligent-Tiering memberikan penghematan biaya secara otomatis dengan memindahkan data pada tingkat objek granular antar tingkat akses ketika pola akses berubah. Untuk data yang dapat diakses secara asinkron, Anda dapat memilih

untuk mengaktifkan pengarsipan otomatis dalam kelas penyimpanan S3 Intelligent-Tiering menggunakanAWS Management Console,AWS CLI, atau Amazon S3 API.

Memindahkan data ke S3 Intelligent-Tiering

Ada dua cara untuk memindahkan data ke dalam S3 Intelligent-Tiering. Anda dapat langsung MEMASUKKAN data ke S3 Intelligent-Tiering dengan menentukan`INTELLIGENT_TIERING` dix-`amz-storage-class` header atau mengonfigurasi konfigurasi Siklus Hidup S3 ke objek transisi dari Standar S3 atau Akses Jarang Standar S3 ke S3 Intelligent-Tiering.

Mengunggah data ke S3 Intelligent-Tiering menggunakan Direct PUT

Saat Anda mengupload objek ke kelas penyimpanan S3 Intelligent-Tiering menggunakan operasi PUT API, Anda menentukan S3 Intelligent-Tiering di header `x-amz-storage-class`permintaan.

Permintaan berikut menyimpan gambar,`my-image.jpg`, dim`yBucket` ember. Permintaan menggunakan`x-amz-storage-class` header untuk meminta agar objek disimpan menggunakan kelas penyimpanan S3 Intelligent-Tiering.

Example

```
PUT /my-image.jpg HTTP/1.1
Host: myBucket.s3.<Region>.amazonaws.com (http://amazonaws.com/)
Date: Wed, 1 Sep 2021 17:50:00 GMT
Authorization: authorization string
Content-Type: image/jpeg
Content-Length: 11434
Expect: 100-continue
x-amz-storage-class: INTELLIGENT_TIERING
```

Transisi data ke S3 Intelligent-Tiering dari S3 Standard atau Standard-Infrequent Access menggunakan S3 Infecycle

Anda dapat menambahkan aturan ke konfigurasi S3 Lifecycle untuk memberi tahu Amazon S3 untuk mentransisikan objek dari satu kelas penyimpanan ke kelas penyimpanan lainnya. Untuk informasi tentang transisi yang didukung dan batasan terkait, lihat Transisi objek menggunakan Siklus Hidup S3.

Anda dapat menentukan konfigurasi S3 Lifecycle pada tingkat awalan atau awalan. Dalam aturan konfigurasi S3 Lifecycle, filter menentukan key prefix (`documents/`). Oleh karena itu, aturan ini

berlaku untuk objek dengan awalan nama kunci `documents/`, seperti `documents/doc1.txt` dan `documents/doc2.txt`. Aturan menentukan`Transition` tindakan mengarahkan mengarahkan mengarahkan mengarahkan mengarahkan Amazon S3 Intelligent-Tiering ke kelas penyimpanan 0 hari setelah pembuatan. Dalam hal ini, objek memenuhi syarat untuk transisi ke S3 Intelligent-Tiering pada tengah malam UTC setelah pembuatan.

Example

```
<LifecycleConfiguration>
  <Rule>
    <ID>ExampleRule</ID>
    <Filter>
       <Prefix>documents/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>0</Days>
      <StorageClass>INTELLIGENT_TIERING</StorageClass>
    </Transition>
 </Rule>
</LifecycleConfiguration>
```

Mengaktifkan Tingkat S3 Intelligent-Archive Access

Untuk mendapatkan biaya penyimpanan terendah pada data yang dapat diakses dalam hitungan menit hingga jam, Anda dapat mengaktifkan salah satu atau kedua tingkatan akses arsip dengan membuat konfigurasi level tag bucket, awalan, atau objek menggunakan APIAWS Management Console,AWS CLI, atau Amazon S3.

Menggunakan konsol S3

Mengaktifkan S3 Intelligent-Tiering

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang Anda inginkan.
3. Pilih Properti.
4. Arahkan ke bagian S3 Intelligent-Tiering Archive configurations dan pilih Create configuration.
5. Di bagian Pengaturan konfigurasi arsip, tentukan nama konfigurasi deskriptif untuk konfigurasi S3 Intelligent-Tiering Archive Anda.

6. Di bawah Pilih cakupan konfigurasi, pilih cakupan konfigurasi yang akan digunakan. Secara opsional, Anda dapat membatasi cakupan konfigurasi ke objek tertentu dalam bucket menggunakan awalan bersama, tag objek, atau kombinasi keduanya.
   a. Untuk membatasi cakupan konfigurasi, pilih Batasi cakupan konfigurasi ini menggunakan satu atau beberapa filter.
   b. Untuk membatasi cakupan konfigurasi menggunakan awalan tunggal, masukkan awalan di bawah Awalan.
   c. Untuk membatasi cakupan konfigurasi menggunakan tag objek, pilih Tambahkan tag dan masukkan nilai untuk Kunci.
7. Di bawah Status, pilih Aktifkan.
8. Di bagian Pengaturan arsip, pilih salah satu atau kedua tingkatan Akses Arsip untuk diaktifkan.
9. Pilih Create (Buat).

Menggunakan AWS CLI

Anda dapat menggunakanAWS CLI perintah berikut untuk mengelola konfigurasi S3 Intelligent-Tiering:

- `delete-bucket-intelligent-tiering-configuration`
- `get-bucket-intelligent-tiering-configuration`
- `list-bucket-intelligent-tiering-configurations`
- `put-bucket-intelligent-tiering-configuration`

Untuk petunjuk tentang pengaturan AWS CLI, lihat Mengembangkan dengan Amazon S3 menggunakan AWS CLI.

Saat menggunakanAWS CLI, Anda tidak dapat menentukan konfigurasi sebagai file XML. Anda harus menentukan JSON sebagai gantinya. Berikut ini adalah contoh konfigurasi XMLS3 Intelligent-Tiering dan JSON setara yang dapat Anda tentukan dalamAWS CLI perintah.

Contoh berikut menempatkan konfigurasi S3 Intelligent-Tiering ke bucket tertentu.

Example `put-bucket-intelligent-tiering-configuration`

JSON

```
{
  "Id": "string",
  "Filter": {
    "Prefix": "string",
    "Tag": {
      "Key": "string",
      "Value": "string"
    },
    "And": {
      "Prefix": "string",
      "Tags": [
        {
          "Key": "string",
          "Value": "string"
        }
        ...
      ]
    }
  },
  "Status": "Enabled"|"Disabled",
  "Tierings": [
    {
      "Days": integer,
      "AccessTier": "ARCHIVE_ACCESS"|"DEEP_ARCHIVE_ACCESS"
    }
    ...
  ]
}
```

XML

```
PUT /?intelligent-tiering&id=Id HTTP/1.1
Host: Bucket.s3.amazonaws.com
<?xml version="1.0" encoding="UTF-8"?>
<IntelligentTieringConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
   <Id>string</Id>
   <Filter>
      <And>
         <Prefix>string</Prefix>
```

```
            <Tag>
               <Key>string</Key>
               <Value>string</Value>
            </Tag>
            ...
         </And>
         <Prefix>string</Prefix>
         <Tag>
            <Key>string</Key>
            <Value>string</Value>
         </Tag>
      </Filter>
      <Status>string</Status>
      <Tiering>
         <AccessTier>string</AccessTier>
         <Days>integer</Days>
      </Tiering>
      ...
   </IntelligentTieringConfiguration>
```

Menggunakan operasi PUT API

Anda dapat menggunakan `PutBucketIntelligentTieringConfiguration`operasi untuk bucket tertentu dan hingga 1.000 konfigurasi Intelligent-Tiering S3 per bucket. Anda dapat menentukan objek dalam bucket mana yang memenuhi syarat untuk tingkat akses Archive menggunakan awalan bersama atau tag objek. Menggunakan awalan bersama atau tag objek memungkinkan Anda menyelaraskan ke aplikasi, alur kerja, atau organisasi internal tertentu. Anda juga memiliki fleksibilitas untuk mengaktifkan tingkat Akses Arsip, tingkatan Deep Archive Access, atau keduanya.

Memulai dengan S3 Intelligent-Tiering

Untuk mempelajari lebih lanjut tentang cara menggunakan S3 Intelligent-Tiering, lihat Tutorial: Memulai menggunakan S3 Intelligent-Tiering.

Mengelola S3 Intelligent-Tiering

Kelas penyimpanan S3 Intelligent-Tiering memberikan penghematan biaya penyimpanan otomatis dalam tiga tingkat akses latensi rendah dan throughput tinggi. Ini juga menawarkan kemampuan arsip opsional untuk membantu Anda mendapatkan biaya penyimpanan terendah di cloud untuk

data yang dapat diakses dalam hitungan menit hingga jam. Kelas penyimpanan S3 Intelligent-Tiering mendukung semua fitur Amazon S3, termasuk yang berikut ini:

- Inventaris S3, untuk memverifikasi tingkat akses objek
- Replikasi S3, untuk mereplikasi data ke apa pun Wilayah AWS
- Lensa Penyimpanan S3, untuk melihat metrik penggunaan dan aktivitas penyimpanan
- Enkripsi di sisi server, untuk melindungi data objek
- Kunci Objek S3, untuk mencegah penghapusan data yang tidak disengaja
- AWS PrivateLink, untuk mengakses Amazon S3 melalui titik akhir pribadi di cloud privat virtual (VPC)

Mengidentifikasi objek tingkat akses S3 Intelligent-Tiering yang disimpan di dalamnya

Untuk mendapatkan daftar objek Anda dan metadata yang sesuai, termasuk tingkat akses S3 Intelligent-Tiering, Anda dapat menggunakannya. the section called “Mengelola inventaris” Inventaris S3 menyediakan file CSV, ORC, atau output Parquet yang mencantumkan objek dan metadata yang sesuai. Anda dapat menerima laporan inventaris ini setiap hari atau setiap pekan untuk bucket Amazon S3 atau awalan bersama. (Awalan bersama mengacu pada objek yang memiliki nama yang dimulai dengan string umum.)

Melihat status arsip objek dalam S3 Intelligent-Tiering

Untuk menerima pemberitahuan ketika objek dalam kelas penyimpanan S3 Intelligent-Tiering telah berpindah ke tingkat Archive Access atau tingkat Deep Archive Access, Anda dapat mengatur Notifikasi Peristiwa S3. Untuk informasi selengkapnya, lihat Mengaktifkan pemberitahuan peristiwa.

Amazon S3 dapat menerbitkan pemberitahuan kejadian ke topik Amazon Simple Notification Service (Amazon SNS), atau fungsi Amazon Simple Queue Service (Amazon SQS), atau fungsi AWS Lambda . Untuk informasi selengkapnya, lihat Notifikasi Peristiwa Amazon S3.

Berikut ini adalah contoh pesan yang dikirimkan Amazon S3 untuk menerbitkan peristiwa `s3:IntelligentTiering`. Untuk informasi selengkapnya, lihat the section called “Struktur pesan peristiwa”.

```
{
   "Records":[
      {
```

```
         "eventVersion":"2.3",
         "eventSource":"aws:s3",
         "awsRegion":"us-west-2",
         "eventTime":"1970-01-01T00:00:00.000Z",
         "eventName":"IntelligentTiering",
         "userIdentity":{
            "principalId":"s3.amazonaws.com"
         },
         "requestParameters":{
            "sourceIPAddress":"s3.amazonaws.com"
         },
         "responseElements":{
            "x-amz-request-id":"C3D13FE58DE4C810",
            "x-amz-id-2":"FMyUVURIY8/IgAtTv8xRjskZQpcIZ9KG4V5Wp6S7S/
JRWeUWerMUE5JgHvANOjpD"
         },
         "s3":{
            "s3SchemaVersion":"1.0",
            "configurationId":"testConfigRule",
            "bucket":{
               "name":"mybucket",
               "ownerIdentity":{
                  "principalId":"A3NL1KOZZKExample"
               },
               "arn":"arn:aws:s3:::mybucket"
            },
            "object":{
               "key":"HappyFace.jpg",
               "size":1024,
               "eTag":"d41d8cd98f00b204e9800998ecf8427e",
            }
         },
         "intelligentTieringEventData":{
            "destinationAccessTier": "ARCHIVE_ACCESS"
            }
      }
   ]
}
```

Anda juga dapat menggunakan Permintaan objek HEAD untuk melihat status arsip objek. Jika objek disimpan di kelas penyimpanan S3 Intelligent-Tiering dan berada di salah satu tingkatan arsip, respons HEAD objek menunjukkan tingkat arsip saat ini. Untuk menampilkan tingkat arsip, permintaan menggunakan header x-amz-archive-status.

Permintaan objek HEAD berikut mengembalikan metadata objek (dalam hal ini, *my-image.jpg*).

Example

```
HEAD /my-image.jpg HTTP/1.1
Host: bucket.s3.region.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:02236Q3V0RonhpaBX5sCYVf1bNRuU=
```

Anda juga dapat menggunakan permintaan objek HEAD untuk memantau status permintaan `restore-object`. Jika pemulihan arsip sedang berlangsung, respons HEAD objek menyertakan header x-amz-restore.

Respons objek HEAD sampel berikut menunjukkan objek yang diarsipkan menggunakan S3 Intelligent-Tiering dengan permintaan pemulihan yang sedang berlangsung.

Example

```
HTTP/1.1 200 OK
x-amz-id-2: FSVaTMjrmBp3Izs1NnwBZeu7M19iI8UbxMbi0A8AirHANJBo+hEftBuiESACOMJp
x-amz-request-id: E5CEFCB143EB505A
Date: Fri, 13 Nov 2020 00:28:38 GMT
Last-Modified: Mon, 15 Oct 2012 21:58:07 GMT
ETag: "1accb31fcf202eba0c0f41fa2f09b4d7"
x-amz-storage-class: 'INTELLIGENT_TIERING'
x-amz-archive-status: 'ARCHIVE_ACCESS'
x-amz-restore: 'ongoing-request="true"'
x-amz-restore-request-date: 'Fri, 13 Nov 2020 00:20:00 GMT'
Accept-Ranges: bytes
Content-Type: binary/octet-stream
Content-Length: 300
Server: AmazonS3
```

Memulihkan objek dari tingkat S3 Intelligent-Tiering Archive Access dan Deep Archive Access

Untuk mengakses objek di tingkat S3 Intelligent-Tiering Archive Access dan Deep Archive Access, Anda harus memulai permintaan pemulihan, dan kemudian menunggu hingga objek dipindahkan ke tingkat Frequent Access. Untuk informasi selengkapnya tentang objek yang diarsipkan, lihatthe section called “Bekerja dengan objek yang diarsipkan”.

Saat memulihkan objek dari tingkat Akses Arsip atau tingkat Akses Arsip Dalam, objek akan kembali ke tingkat Akses Sering. Setelah itu, jika objek tidak diakses selama 30 hari berturut-turut, objek akan secara otomatis berpindah ke tingkat Infrequent Access. Kemudian, setelah minimal 90 hari berturut-turut tanpa akses, objek berpindah ke tingkat Archive Access. Setelah minimal 180 hari berturut-turut tanpa akses, objek berpindah ke tingkat Deep Archive Access. Untuk informasi selengkapnya, lihat the section called “Cara kerja S3 Intelligent-Tiering”.

Anda dapat memulihkan objek yang diarsipkan menggunakan konsol Amazon S3, Operasi Batch S3, API REST Amazon S3, SDK, AWS atau (). AWS Command Line Interface AWS CLI Untuk informasi selengkapnya, lihat the section called “Bekerja dengan objek yang diarsipkan”.

Mengelola siklus hidup penyimpanan Anda

Untuk mengelola objek agar disimpan secara efektif sepanjang siklus hidupnya, buat konfigurasi Siklus Hidup Amazon S3. Konfigurasi Siklus Hidup Amazon S3 adalah seperangkat aturan yang menentukan tindakan yang diterapkan Amazon S3 ke sekelompok objek. Ada dua jenis tindakan:

- Tindakan transisi–Tindakan ini menentukan kapan objek bertransisi ke kelas penyimpanan lainnya. Misalnya, Anda dapat memilih untuk melakukan transisi objek ke kelas penyimpanan S3 Standard-IA dalam waktu 30 hari setelah membuatnya, atau mengarsipkan objek ke kelas penyimpanan S3 Glacier Flexible Retrieval satu tahun setelah membuatnya. Untuk informasi selengkapnya, lihat Menggunakan kelas penyimpanan Amazon S3.

  Ada biaya yang terkait dengan permintaan transisi siklus hidup. Untuk informasi harga, lihat Harga Amazon S3.
- Tindakan kedaluwarsa–Tindakan ini menentukan kapan objek kedaluwarsa. Amazon S3 menghapus objek kedaluwarsa atas nama Anda.

  Biaya kedaluwarsa siklus hidup bergantung pada saat Anda memilih untuk mengakhiri objek. Untuk informasi selengkapnya, lihat Mengakhiri objek.

> **Important**
>
> Anda tidak dapat menggunakan kebijakan bucket untuk mencegah penghapusan atau transisi oleh aturan Siklus Hidup S3. Misalnya, meskipun kebijakan bucket Anda menolak semua tindakan untuk semua prinsipal, konfigurasi Siklus Hidup S3 Anda tetap berfungsi seperti biasa.

Objek yang ada dan baru

Ketika Anda menambahkan konfigurasi Siklus Hidup ke dalam bucket, aturan konfigurasi berlaku untuk objek yang ada dan objek yang Anda tambahkan kemudian. Misalnya, jika Anda menambahkan aturan konfigurasi Siklus Hidup hari ini dengan tindakan kedaluwarsa yang menyebabkan objek kedaluwarsa 30 hari setelah pembuatan, Amazon S3 akan mengantri untuk menghapus objek yang ada yang berumur lebih dari 30 hari.

Perubahan dalam penagihan

Jika ada penundaan antara saat objek memenuhi syarat untuk tindakan siklus hidup dan saat Amazon S3 mentransfer atau mengakhiri objek Anda, perubahan penagihan akan diterapkan segera setelah objek memenuhi syarat untuk tindakan siklus hidup. Misalnya, jika objek dijadwalkan kedaluwarsa dan Amazon S3 tidak segera kedaluwarsa objek, Anda tidak akan dikenakan biaya untuk penyimpanan setelah waktu kedaluwarsa.

Satu pengecualian untuk perilaku ini adalah jika Anda memiliki aturan siklus hidup untuk bertransisi ke kelas penyimpanan S3 Intelligent-Tiering. Dalam hal ini, perubahan penagihan tidak terjadi hingga objek telah ditransisikan ke S3 Intelligent-Tiering.

Untuk informasi selengkapnya tentang aturan Siklus Hidup S3, lihat Elemen konfigurasi Siklus Hidup.

Memantau pengaruh aturan siklus hidup

Untuk memantau efek pembaruan yang dibuat oleh aturan siklus hidup aktif, lihat. the section called “Bagaimana cara memantau tindakan yang diambil oleh aturan siklus hidup saya?”

Mengelola siklus hidup objek

Tetapkan aturan konfigurasi Siklus Hidup S3 untuk objek dengan siklus hidup yang didefinisikan dengan baik. Sebagai contoh:

- Jika Anda mengunggah log berkala ke bucket, aplikasi Anda mungkin memerlukannya selama seminggu atau sebulan. Setelah itu, Anda mungkin ingin menghapusnya.
- Beberapa dokumen sering diakses untuk periode waktu terbatas. Setelah itu, dokumen tersebut jarang diakses. Sewaktu-waktu, Anda mungkin tidak memerlukan akses waktu nyata, tetapi organisasi atau peraturan Anda mungkin mengharuskan Anda untuk mengarsipkannya selama periode tertentu. Setelah itu, Anda dapat menghapusnya.
- Anda dapat mengunggah beberapa jenis data ke Amazon S3 terutama untuk tujuan pengarsipan. Misalnya, Anda dapat mengarsipkan media digital, catatan keuangan dan kesehatan, data urutan

genomik mentah, pencadangan basis data jangka panjang, dan data yang harus disimpan untuk kepatuhan terhadap peraturan.

Dengan aturan konfigurasi Siklus Hidup S3, Anda dapat memberi tahu Amazon S3 untuk melakukan transisi objek ke kelas penyimpanan yang lebih murah, atau mengarsipkan, atau menghapusnya.

Membuat konfigurasi siklus hidup

Konfigurasi Siklus Hidup S3 merupakan file XML yang terdiri dari serangkaian aturan dengan tindakan yang telah ditentukan sebelumnya yang Anda ingin Amazon S3 lakukan pada objek selama masa hidupnya.

Anda dapat membuat konfigurasi siklus hidup menggunakan konsol Amazon S3, REST API, SDK AWS , dan (). AWS Command Line Interface AWS CLI Untuk informasi selengkapnya, lihat Menyetel konfigurasi siklus hidup pada bucket.

Amazon S3 menyediakan serangkaian operasi API REST untuk mengelola konfigurasi siklus hidup pada bucket. Amazon S3 menyimpan konfigurasi sebagai subsumber daya siklus hidup yang dilampirkan ke bucket Anda. Untuk detailnya, lihat berikut ini:

- PutBucketLifecycleConfiguration
- GetBucketLifecycleConfiguration
- DeleteBucketLifecycle

Untuk informasi selengkapnya tentang membuat konfigurasi siklus hidup, lihat topik berikut ini:

Topik

- Transisi objek menggunakan Siklus Hidup Amazon S3
- Mengakhiri objek
- Menyetel konfigurasi siklus hidup pada bucket
- Siklus hidup dan konfigurasi bucket lainnya
- Mengonfigurasi notifikasi peristiwa Siklus Hidup
- Elemen konfigurasi Siklus Hidup
- Contoh konfigurasi Siklus Hidup S3

Transisi objek menggunakan Siklus Hidup Amazon S3

Anda dapat menambahkan aturan dalam konfigurasi Siklus Hidup S3 guna memberi tahu Amazon S3 untuk melakukan transisi objek ke kelas penyimpanan Amazon S3 lain. Untuk informasi selengkapnya tentang kelas penyimpanan, lihat Menggunakan kelas penyimpanan Amazon S3. Beberapa contoh kapan Anda dapat menggunakan konfigurasi Siklus Hidup S3 dengan cara ini termasuk berikut ini:

- Ketika Anda mengetahui bahwa objek jarang diakses, Anda dapat mengalihkannya ke kelas penyimpanan S3 Standard-IA.
- Anda mungkin ingin mengarsipkan objek yang tidak perlu Anda akses secara real time ke kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive.

Objek yang ada dan baru

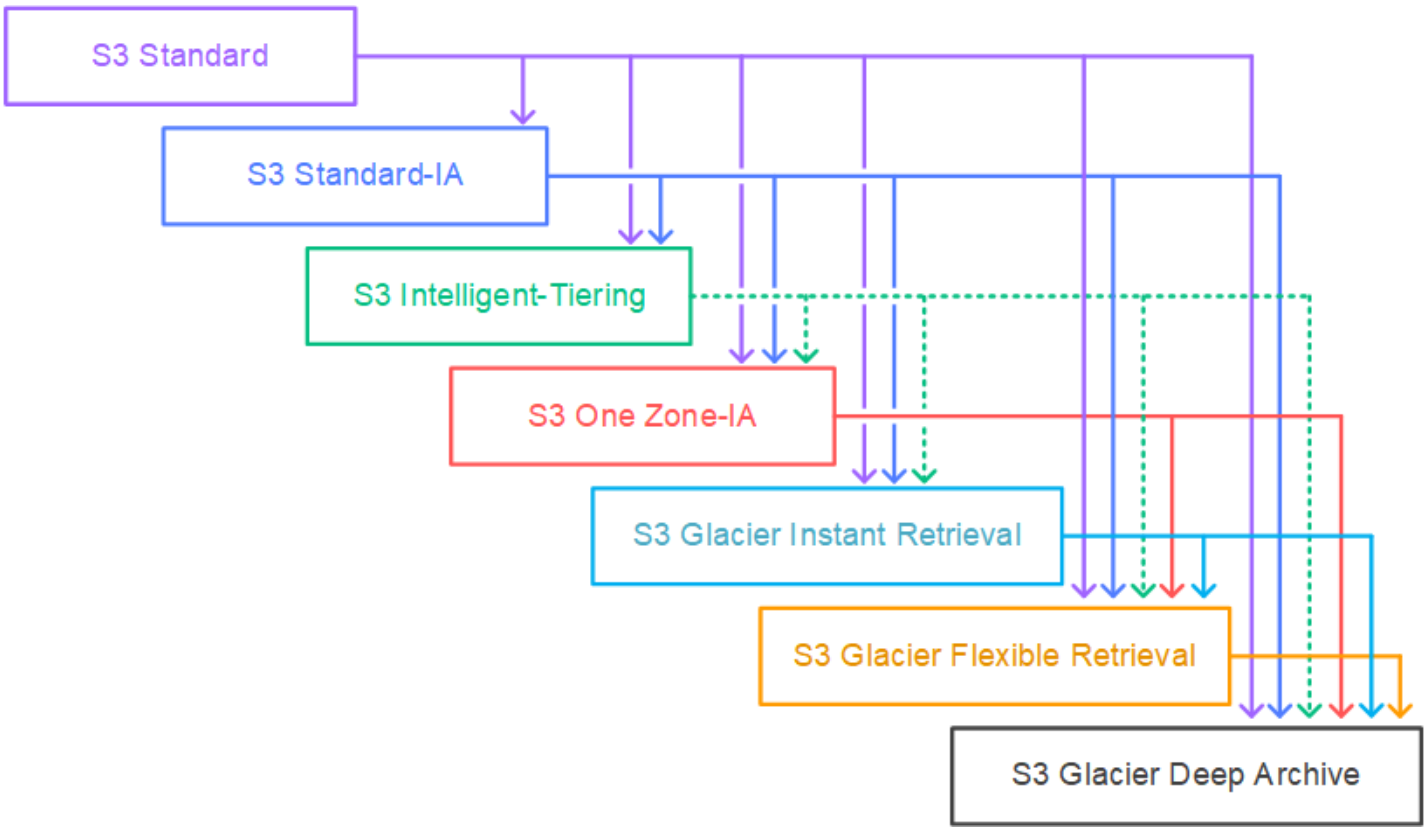
Ketika Anda menambahkan konfigurasi Siklus Hidup ke dalam bucket, aturan konfigurasi berlaku untuk objek yang ada dan objek yang Anda tambahkan kemudian. Misalnya, jika Anda menambahkan aturan konfigurasi Siklus Hidup hari ini dengan tindakan transisi yang menyebabkan objek dengan awalan tertentu bertransisi ke kelas penyimpanan yang berbeda 30 hari setelah pembuatan, Amazon S3 akan mengantri untuk transisi objek yang ada yang berusia lebih dari 30 hari dan yang memiliki awalan yang ditentukan.

> **Important**
>
> Anda tidak dapat menggunakan kebijakan bucket untuk mencegah penghapusan atau transisi oleh aturan Siklus Hidup S3. Misalnya, meskipun kebijakan bucket Anda menolak semua tindakan untuk semua prinsipal, konfigurasi Siklus Hidup S3 Anda tetap berfungsi seperti biasa.

Transisi yang didukung dan kendala terkait

Dalam konfigurasi Siklus Hidup S3, Anda dapat menentukan aturan untuk transisi objek dari satu kelas penyimpanan ke kelas penyimpanan lainnya guna menghemat biaya penyimpanan. Saat Anda tidak mengetahui pola akses objek Anda, atau pola akses Anda berubah seiring waktu berjalan, Anda dapat melakukan transisi objek ke kelas penyimpanan S3 Intelligent-Tiering untuk penghematan biaya otomatis. Untuk informasi tentang kelas penyimpanan, lihat Menggunakan kelas penyimpanan Amazon S3.

Amazon S3 mendukung model air terjun untuk transisi antar kelas penyimpanan, seperti yang ditunjukkan pada diagram berikut.



Transisi siklus hidup yang didukung

Amazon S3 mendukung transisi siklus hidup antara kelas penyimpanan berikut menggunakan konfigurasi Siklus Hidup S3.

Anda dapat melakukan transisi dari berikut ini:

- Kelas penyimpanan S3 Standard ke kelas penyimpanan lainnya.
- Kelas penyimpanan S3 Standard-IA ke kelas penyimpanan S3 Intelligent-Tiering, S3 One Zone-IA, S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval, atau S3 Glacier Deep Archive.
- Kelas penyimpanan S3 Intelligent-Tiering ke kelas penyimpanan S3 One Zone-IA, S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval, atau S3 Glacier Deep Archive.

> **Note**
>
> Ada beberapa pengecualian untuk transisi objek dari kelas penyimpanan S3 Intelligent-Tiering ke S3 One Zone-IA dan beberapa kelas penyimpanan S3 Glacier. Untuk informasi selengkapnya, lihat the section called "Transisi siklus hidup yang tidak didukung".

- Kelas penyimpanan S3 One Zone-IA ke kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive.
- Kelas penyimpanan S3 Glacier Instant Retrieval ke kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive.
- Kelas penyimpanan S3 Glacier Flexible Retrieval ke kelas penyimpanan S3 Glacier Deep Archive.
- Kelas penyimpanan apa pun ke kelas penyimpanan S3 Glacier Deep Archive.

> **Note**
>
> Tidak ada biaya pengambilan data untuk transisi siklus hidup. Namun, ada biaya konsumsi per permintaan saat menggunakan`PUT`,`COPY`, atau aturan siklus hidup untuk memindahkan data ke kelas penyimpanan S3 apa pun. Pertimbangkan biaya konsumsi atau transisi sebelum memindahkan objek ke kelas penyimpanan apa pun. Untuk informasi selengkapnya tentang pertimbangan biaya, lihat Harga Amazon S3.

Transisi siklus hidup yang tidak didukung

Amazon S3 tidak mendukung transisi siklus hidup berikut.

Anda tidak dapat melakukan transisi dari berikut ini:

- Kelas penyimpanan apa pun ke kelas penyimpanan S3 Standard.
- Kelas penyimpanan apa pun ke kelas Reduced Redundancy Storage (RRS).
- Kelas penyimpanan S3 One Zone-IA ke kelas penyimpanan S3 Intelligent-Tiering, S3 Standard-IA, atau S3 Glacier Instant Retrieval.
- Kelas penyimpanan S3 Intelligent-Tiering (semua tingkatan) ke kelas penyimpanan IA Standar S3.
- Kelas penyimpanan S3 Intelligent-Tiering Arsipkan tingkat Akses Instan ke S3 One Zone-IA.

- Tingkat Akses Arsip kelas penyimpanan S3 Intelligent-Tiering ke Pengambilan Instan S3 One Zone-IA atau S3 Glacier.
- Kelas penyimpanan S3 Intelligent-Tiering Tingkat Akses Arsip Dalam ke S3 One Zone-IA, S3 Glacier Instant Retrieval, atau S3 Glacier Flexible Retrieval.

Batasan

Transisi kelas penyimpanan siklus hidup memiliki kendala berikut:

Ukuran dan Transisi Objek dari S3 Standard atau S3 Standard-IA ke S3 Intelligent-Tiering, S3 Standard-IA, atau S3 One Zone-IA

Ketika Anda melakukan transisi objek dari kelas penyimpanan S3 Standard atau S3 Standard-IA ke S3 Intelligent-Tiering, S3 Standard-IA, atau S3 One Zone-IA, pembatasan ukuran objek berikut berlaku:

- Objek yang lebih besar–Untuk transisi berikut, ada keuntungan biaya untuk transisi objek yang lebih besar:
  - Dari kelas penyimpanan S3 Standard atau S3 Standard-IA ke S3 Intelligent-Tiering.
  - Dari kelas penyimpanan S3 Standard ke S3 Standard-IA atau S3 One Zone-IA.
- Objek yang lebih kecil dari 128 KiB — Untuk transisi berikut, Amazon S3 tidak mentransisikan objek yang lebih kecil dari 128 KiB:
  - Dari kelas penyimpanan S3 Standard atau S3 Standard-IA ke S3 Intelligent-Tiering atau S3 Glacier Instant Retrieval.
  - Dari kelas penyimpanan S3 Standard ke S3 Standard-IA atau S3 One Zone-IA.

> **Note**
>
> Anda dapat memfilter aturan siklus hidup berdasarkan ukuran objek.

> **Important**
>
> Jika Anda memiliki beberapa aturan dalam konfigurasi Siklus Hidup S3, objek dapat memenuhi syarat untuk beberapa tindakan Siklus Hidup S3 pada hari yang sama. Dalam kasus tersebut, Amazon S3 mengikuti aturan umum ini:

- Penghapusan permanen lebih diutamakan daripada transisi.
- Transisi lebih diutamakan daripada pembuatan penanda hapus.
- Saat sebuah objek memenuhi syarat untuk transisi S3 Glacier Flexible Retrieval dan S3 Standard-IA (atau S3 One Zone-IA), Amazon S3 memilih transisi S3 Glacier Flexible Retrieval.

Sebagai contoh, lihat Contoh 5: Filter yang tumpang tindih, tindakan siklus hidup yang bertentangan, dan apa yang dilakukan Amazon S3 dengan bucket tanpa versi.

Hari Minimum untuk Transisi ke S3 Standard-IA atau S3 One Zone-IA

Sebelum Anda melakukan transisi objek ke S3 Standard-IA atau S3 One Zone-IA, Anda harus menyimpannya setidaknya 30 hari di Amazon S3. Misalnya, Anda tidak dapat membuat aturan Siklus Hidup untuk memindahkan objek ke kelas penyimpanan S3 Standard-IA satu hari setelah Anda membuatnya. Amazon S3 tidak mendukung transisi ini dalam 30 hari pertama karena objek yang lebih baru sering diakses atau dihapus lebih cepat daripada yang sesuai untuk penyimpanan S3 Standard-IA atau S3 One Zone-IA.

Demikian pula, jika Anda sedang melakukan transisi objek lama (dalam bucket berversi), Anda hanya dapat melakukan transisi objek yang setidaknya 30 hari lamanya ke penyimpanan S3 Standard-IA atau S3 One Zone-IA. Untuk daftar durasi penyimpanan minimum untuk semua kelas penyimpanan, lihatMembandingkan kelas penyimpanan Amazon S3.

Biaya Penyimpanan Minimum 30 Hari untuk S3 Standard-IA dan S3 One Zone-IA

Kelas penyimpanan S3 Standard-IA dan S3 One Zone-IA memiliki biaya penyimpanan minimum 30 hari. Oleh karena itu, Anda tidak dapat menentukan satu aturan Siklus Hidup untuk transisi S3 Standard-IA atau S3 One Zone-IA dan transisi S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive saat transisi S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive terjadi kurang dari 30 hari setelah transisi S3 Standard-IA atau S3 One Zone-IA.

30 hari minimum yang sama berlaku saat Anda menentukan transisi dari penyimpanan S3 Standard-IA ke S3 One Zone-IA. Anda dapat menentukan dua aturan untuk mencapai hal ini, tetapi Anda membayar biaya penyimpanan minimum. Untuk informasi selengkapnya tentang pertimbangan biaya, lihat Harga Amazon S3.

Mengelola siklus hidup lengkap objek

Anda dapat menggabungkan tindakan Siklus Hidup S3 ini untuk mengelola siklus hidup lengkap sebuah objek. Misalnya, bayangkan objek yang Anda buat memiliki siklus hidup yang didefinisikan dengan baik. Pada awalnya, objek sering diakses selama periode 30 hari. Kemudian, objek jarang diakses hingga 90 hari. Setelah itu, objek tidak lagi diperlukan, jadi Anda dapat memilih untuk mengarsipkan atau menghapusnya.

Dalam skenario ini, Anda dapat membuat aturan Siklus Hidup S3, yaitu Anda menentukan tindakan transisi awal ke penyimpanan S3 Intelligent-Tiering, S3 Standard-IA, atau S3 One Zone-IA, tindakan transisi lain ke penyimpanan S3 Glacier Flexible Retrieval untuk pengarsipan, dan tindakan kedaluwarsa. Saat Anda memindahkan objek dari satu kelas penyimpanan ke kelas penyimpanan lainnya, Anda menghemat biaya penyimpanan. Untuk informasi selengkapnya tentang pertimbangan biaya, lihat Harga Amazon S3.

Transisi ke kelas penyimpanan S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive (pengarsipan objek)

Dengan menggunakan konfigurasi Siklus Hidup S3, Anda dapat mentransisikan objek ke kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive untuk pengarsipan. Saat memilih kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, objek akan tetap berada di dalam Amazon S3. Anda tidak dapat mengaksesnya secara langsung melalui layanan Amazon S3 Glacier yang terpisah. Untuk informasi lebih umum tentang S3 Glacier lihat, Apa itu Amazon S3 Glacier di Panduan Developer Amazon S3 Glacier.

Sebelum mengarsipkan objek, pelajari bagian berikut untuk pertimbangan yang relevan.

Pertimbangan umum

Berikut ini adalah pertimbangan umum yang perlu Anda pertimbangkan sebelum mengarsipkan objek:

- Objek terenkripsi tetap dienkripsi selama proses transisi kelas penyimpanan.
- Objek yang disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive tidak tersedia secara waktu nyata.

  Objek yang diarsipkan adalah objek Amazon S3, tetapi sebelum Anda dapat mengakses objek yang diarsipkan, Anda harus memulihkan salinan sementara terlebih dahulu. Salinan objek yang dipulihkan hanya tersedia selama durasi yang Anda tentukan dalam permintaan pemulihan.

Setelah itu, Amazon S3 menghapus salinan sementara, dan objek tetap diarsipkan di S3 Glacier Flexible Retrieval.

Anda dapat memulihkan objek dengan menggunakan konsol Amazon S3 atau secara terprogram dengan menggunakan pustaka pembungkus AWS SDK atau Amazon S3 REST API dalam kode Anda. Untuk informasi selengkapnya, lihat Memulihkan objek yang diarsipkan.

- Objek yang disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval hanya dapat dialihkan ke kelas penyimpanan S3 Glacier Deep Archive.

  Anda dapat menggunakan aturan konfigurasi Siklus Hidup S3 untuk mengonversi kelas penyimpanan objek dari S3 Glacier Flexible Retrieval ke kelas penyimpanan S3 Glacier Deep Archive saja. Jika Anda ingin mengubah kelas penyimpanan objek yang disimpan dalam S3 Glacier Flexible Retrieval ke kelas penyimpanan selain S3 Glacier Deep Archive, Anda harus menggunakan operasi pemulihan untuk membuat salinan sementara dari objek terlebih dahulu. Kemudian gunakan operasi penyalinan untuk menimpa objek yang menentukan S3 Standard, S3 Intelligent-Tiering, S3 Standard-IA, S3 One Zone-IA, atau Reduced Redundancy sebagai kelas penyimpanan.
- Transisi objek ke kelas penyimpanan S3 Glacier Deep Archive hanya dapat menggunakan satu cara.

  Anda tidak dapat menggunakan aturan konfigurasi Siklus Hidup S3 untuk mengonversi kelas penyimpanan objek dari S3 Glacier Deep Archive ke kelas penyimpanan lainnya. Jika Anda ingin mengubah kelas penyimpanan dari objek yang diarsipkan ke kelas penyimpanan lain, Anda harus menggunakan operasi pemulihan untuk membuat salinan sementara dari objek terlebih dahulu. Kemudian gunakan operasi penyalinan untuk menimpa objek yang menentukan S3 Standard, S3 Intelligent-Tiering, S3 Standard-IA, S3 One Zone-IA, S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval, atau Reduced Redundancy Storage sebagai kelas penyimpanan.

> **Note**
>
> Operasi Salin untuk objek yang dipulihkan tidak didukung di konsol S3 Amazon untuk objek di kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive. Untuk jenis operasi Copy ini, gunakan AWS Command Line Interface (AWS CLI), AWS SDK, atau REST API.

Objek yang disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive hanya dapat dilihat dan tersedia melalui Amazon S3. Objek ini tidak tersedia melalui layanan Amazon S3 Glacier yang terpisah.

Ini adalah objek Amazon S3, dan Anda dapat mengaksesnya hanya dengan menggunakan konsol Amazon S3 atau API Amazon S3. Anda tidak dapat mengakses objek yang diarsipkan melalui konsol Amazon S3 Glacier terpisah atau API Amazon S3 Glacier.

Pertimbangan biaya

Jika Anda berencana untuk mengarsipkan data yang jarang diakses selama beberapa bulan atau tahun, kelas penyimpanan S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive dapat mengurangi biaya penyimpanan Anda. Namun, untuk memastikan bahwa kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive sesuai bagi Anda, pertimbangkan hal berikut ini:

- Biaya overhead penyimpanan–Saat Anda melakukan transisi objek ke kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, jumlah penyimpanan tetap ditambahkan ke setiap objek untuk mengakomodasi metadata pengelolaan objek.
  - Untuk setiap objek yang diarsipkan ke S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, Amazon S3 menggunakan penyimpanan 8 KB untuk nama objek dan metadata lainnya. Amazon S3 menyimpan metadata ini sehingga Anda bisa mendapatkan daftar objek yang diarsipkan secara waktu nyata dengan menggunakan API Amazon S3. Untuk informasi selengkapnya, lihat Get Bucket (Daftar Objek). Anda dikenakan tarif Standar S3 untuk penyimpanan tambahan ini.
  - Untuk setiap objek yang diarsipkan ke S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, Amazon S3 menambahkan penyimpanan 32 KB untuk indeks dan metadata terkait. Data ekstra ini diperlukan untuk mengidentifikasi dan memulihkan objek Anda. Anda dikenakan tarif S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive untuk penyimpanan tambahan ini.

  Jika Anda mengarsipkan objek kecil, pertimbangkan biaya penyimpanan ini. Pertimbangkan juga untuk menggabungkan banyak objek kecil menjadi sejumlah kecil objek besar untuk mengurangi biaya overhead.
- Jumlah hari yang Anda rencanakan untuk menyimpan objek yang diarsipkan–S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive adalah solusi pengarsipan jangka panjang. Periode durasi

penyimpanan minimal adalah 90 hari untuk kelas penyimpanan S3 Glacier Flexible Retrieval dan 180 hari untuk S3 Glacier Deep Archive. Menghapus data yang diarsipkan ke Amazon S3 Glacier tidak dikenakan biaya jika objek yang Anda hapus diarsipkan lebih dari periode durasi penyimpanan minimal. Jika Anda menghapus atau menimpa objek yang diarsipkan dalam periode durasi minimal, Amazon S3 mengenakan biaya penghapusan dini prorata. Untuk informasi tentang biaya penghapusan dini, lihat “Bagaimana saya dikenakan biaya untuk menghapus objek dari Amazon S3 Glacier yang berumur kurang dari 90 hari?” pertanyaan di FAQ Amazon S3.

- Biaya permintaan transisi S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive–Masing-masing objek yang Anda transisikan ke kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive merupakan satu permintaan transisi. Ada biaya untuk setiap permintaan tersebut. Jika Anda berencana untuk mentransisikan banyak objek, pertimbangkan biaya permintaan. Jika Anda mengarsipkan campuran objek yang menyertakan objek kecil, terutama yang di bawah 128KB, sebaiknya gunakan filter ukuran objek siklus hidup untuk menyaring objek kecil dari transisi Anda guna mengurangi biaya permintaan. S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive tidak secara otomatis memblokir transisi objek di bawah 128KB.
- Biaya pemulihan data S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive–S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive dirancang untuk pengarsipan data jangka panjang yang jarang Anda akses. Untuk informasi tentang biaya pemulihan data, lihat “Berapa biaya untuk mengambil data dari Amazon S3 Glacier?” pertanyaan di FAQ Amazon S3. Untuk informasi tentang cara memulihkan data dari Amazon S3 Glacier, lihat Memulihkan objek yang diarsipkan.

Saat Anda mengarsipkan objek ke Amazon S3 Glacier dengan menggunakan manajemen Siklus Hidup S3, Amazon S3 mentransisi objek ini secara asinkron. Mungkin ada penundaan antara tanggal transisi dalam aturan konfigurasi Siklus Hidup S3 dan tanggal transisi fisik. Anda dikenakan harga Amazon S3 Glacier berdasarkan tanggal transisi yang ditentukan dalam aturan. Untuk informasi selengkapnya, lihat bagian Amazon S3 Glacier dari FAQ Amazon S3.

Halaman detail produk Amazon S3 menyediakan informasi harga dan contoh perhitungan untuk mengarsipkan objek Amazon S3. Untuk informasi selengkapnya, lihat topik berikut:

- “Bagaimana biaya penyimpanan saya dihitung untuk objek Amazon S3 yang diarsipkan ke Amazon S3 Glacier?” di FAQ Amazon S3.
- “Bagaimana saya dikenakan biaya untuk menghapus objek dari Amazon S3 Glacier yang berumur kurang dari 90 hari?” di FAQ Amazon S3.
- “Berapa biaya untuk mengambil data dari Amazon S3 Glacier?” di FAQ Amazon S3.
- Harga Amazon S3 untuk biaya penyimpanan untuk berbagai kelas penyimpanan.

Memulihkan objek yang diarsipkan

Objek yang diarsipkan tidak dapat diakses secara real time. Anda harus memulai permintaan pemulihan terlebih dahulu, kemudian menunggu hingga salinan sementara objek tersedia selama durasi yang Anda tentukan dalam permintaan. Setelah Anda menerima salinan sementara dari objek yang dipulihkan, kelas penyimpanan objek tetaplah S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive. (Permintaan operasi A HeadObjectatau GetObjectAPI akan mengembalikan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive sebagai kelas penyimpanan.)

> **Note**
>
> Saat memulihkan arsip, Anda membayar untuk arsip tersebut (S3 Glacier Flexible Retrieval atau tingkat S3 Glacier Deep Archive) dan salinan yang Anda pulihkan sementara (tingkat penyimpanan S3 Standard). Untuk informasi selengkapnya tentang harga, silakan lihat Harga Amazon S3.

Anda dapat memulihkan salinan objek secara terprogram atau dengan menggunakan konsol Amazon S3. Amazon S3 hanya memproses satu permintaan pemulihan pada satu waktu per objek. Untuk informasi selengkapnya, lihat Memulihkan objek yang diarsipkan.

Mengakhiri objek

Saat objek mencapai akhir masa pakainya berdasarkan konfigurasi siklus hidupnya, Amazon S3 mengambil tindakan berdasarkan status Pembuatan Versi S3 tempat bucket berada.

- Bucket nonversioned — Amazon S3 mengantri objek untuk dihapus dan menghapusnya secara asinkron, menghapus objek secara permanen.
- Bucket dengan dukungan Penentuan Versi–Jika versi objek saat ini bukan penanda hapus, Amazon S3 menambahkan penanda hapus dengan ID versi unik. Ini membuat versi saat ini menjadi versi lama, dan penanda hapus menjadi versi saat ini.
- Bucket dengan Penentuan Versi ditangguhkan–Amazon S3 membuat penanda hapus dengan null sebagai ID versi. Penanda hapus ini menggantikan versi objek apa pun dengan ID versi null dalam hierarki versi, yang secara efektif menghapus objek.

Untuk bucket berversi (yaitu, dengan dukungan Penentuan Versi atau dengan Penentuan Versi ditangguhkan), ada beberapa pertimbangan yang memandu cara Amazon S3 menangani tindakan

Kedaluwarsa. Untuk bucket dengan dukungan Penentuan Versi atau Penentuan Versi ditangguhkan, berlaku berikut ini:

- Kedaluwarsa objek hanya berlaku untuk versi objek saat ini (tidak memiliki dampak pada versi objek lama).
- Amazon S3 tidak mengambil tindakan apa pun jika ada satu atau beberapa versi objek dan penanda hapus adalah versi saat ini.
- Jika versi objek saat ini adalah satu-satunya versi objek dan juga merupakan penanda hapus (juga disebut sebagai penanda hapus objek kedaluwarsa, ketika semua versi objek dihapus dan Anda hanya memiliki penanda hapus yang tersisa), Amazon S3 menghapus penanda hapus objek kedaluwarsa. Anda juga dapat menggunakan tindakan kedaluwarsa guna mengarahkan Amazon S3 untuk menghapus penanda hapus objek yang kedaluwarsa. Sebagai contoh, lihat Contoh 7: Menghapus penanda hapus objek kedaluwarsa.
- Anda dapat menggunakan elemen `NoncurrentVersionExpiration` tindakan untuk mengarahkan Amazon S3 untuk menghapus versi objek noncurrent secara permanen. Objek yang dihapus ini tidak dapat dipulihkan. Anda dapat mendasarkan kedaluwarsa ini pada sejumlah hari tertentu sejak objek menjadi tidak aktif. Selain jumlah hari, Anda juga dapat memberikan jumlah maksimum versi noncurrent untuk dipertahankan (antara 1 dan 100). Nilai ini menentukan berapa banyak versi lama yang harus ada sebelum Amazon S3 dapat melakukan tindakan terkait pada versi tertentu. Untuk menentukan jumlah maksimum versi noncurrent, Anda juga harus menyediakan `Filter` elemen. Jika Anda tidak menentukan `Filter` elemen, Amazon S3 menghasilkan `InvalidRequest` kesalahan saat Anda memberikan jumlah maksimum versi noncurrent. Untuk informasi selengkapnya tentang menggunakan elemen `NoncurrentVersionExpiration` tindakan, lihatthe section called “Elemen untuk mendeskripsikan tindakan siklus hidup”.

Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.

> **Important**
>
> Jika Anda memiliki beberapa aturan dalam konfigurasi Siklus Hidup S3, objek dapat memenuhi syarat untuk beberapa tindakan Siklus Hidup S3 pada hari yang sama. Dalam kasus tersebut, Amazon S3 mengikuti aturan umum ini:
>
> - Penghapusan permanen lebih diutamakan daripada transisi.
> - Transisi lebih diutamakan daripada pembuatan penanda hapus.

- Ketika objek memenuhi syarat untuk transisi S3 Glacier Flexible Retrieval dan S3 Standard-IA (atau S3 One Zone-IA) transisi, Amazon S3 memilih transisi S3 Glacier Flexible Retrieval.

Sebagai contoh, lihat Contoh 5: Filter yang tumpang tindih, tindakan siklus hidup yang bertentangan, dan apa yang dilakukan Amazon S3 dengan bucket tanpa versi.

Objek yang ada dan baru

Ketika Anda menambahkan konfigurasi Siklus Hidup ke dalam bucket, aturan konfigurasi berlaku untuk objek yang ada dan objek yang Anda tambahkan kemudian. Misalnya, jika Anda menambahkan aturan konfigurasi Siklus Hidup hari ini dengan tindakan kedaluwarsa yang menyebabkan objek dengan awalan tertentu kedaluwarsa 30 hari setelah pembuatan, Amazon S3 akan mengantri untuk menghapus objek yang ada yang berusia lebih dari 30 hari dan yang memiliki awalan yang ditentukan.

Important

Anda tidak dapat menggunakan kebijakan bucket untuk mencegah penghapusan atau transisi oleh aturan Siklus Hidup S3. Misalnya, meskipun kebijakan bucket Anda menolak semua tindakan untuk semua prinsipal, konfigurasi Siklus Hidup S3 Anda tetap berfungsi seperti biasa.

Cara mengetahui kapan objek akan kedaluwarsa

Untuk mengetahui kapan objek dijadwalkan kedaluwarsa, gunakan operasi HeadObjectatau GetObjectAPI. Operasi API ini menampilkan respons header yang memberikan tanggal dan waktu ketika objek tidak lagi dapat di-cache.

Note

- Mungkin ada penundaan antara tanggal kedaluwarsa dan tanggal ketika Amazon S3 menghapus objek. Anda tidak dikenakan biaya kedaluwarsa atau waktu penyimpanan yang terkait dengan objek yang telah kedaluwarsa.

- Sebelum memperbarui, menonaktifkan, atau menghapus aturan Siklus Hidup, gunakan operasi `LIST` API (seperti, ListObjectsV2ListObjectVersions, dan ListMultipartUploads) atau untuk memverifikasi Inventaris Amazon S3 bahwa Amazon S3 telah mentransisikan dan menghapus objek yang memenuhi syarat berdasarkan kasus penggunaan Anda.

Biaya durasi penyimpanan minimum

Jika Anda membuat aturan kedaluwarsa Siklus Hidup S3 yang menyebabkan objek dalam penyimpanan S3 Standard-IA atau S3 One Zone-IA selama kurang dari 30 hari kedaluwarsa, Anda akan dikenakan biaya selama 30 hari. Jika Anda membuat aturan kedaluwarsa Siklus Hidup yang menyebabkan objek dalam penyimpanan S3 Glacier Flexible Retrieval selama kurang dari 90 hari kedaluwarsa, Anda akan dikenakan biaya selama 90 hari. Jika Anda membuat aturan kedaluwarsa Siklus Hidup yang menyebabkan objek dalam penyimpanan S3 Glacier Deep Archive selama kurang dari 180 hari kedaluwarsa, Anda akan dikenakan biaya selama 180 hari.

Untuk informasi selengkapnya, lihat Harga Amazon S3.

Menyetel konfigurasi siklus hidup pada bucket

Bagian ini menjelaskan cara menyetel konfigurasi Siklus Hidup Amazon S3 pada bucket dengan menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), SDK AWS , atau Amazon S3 REST API. Untuk informasi tentang konfigurasi Siklus Hidup S3, lihat Mengelola siklus hidup penyimpanan Anda.

Anda dapat menggunakan aturan siklus hidup untuk menentukan tindakan yang Anda ingin dilakukan Amazon S3 selama masa pakai objek (misalnya, objek transisi ke kelas penyimpanan lain, mengarsipkannya, atau menghapusnya setelah periode waktu tertentu).

Sebelum Anda mengatur konfigurasi siklus hidup, perhatikan hal berikut ini:

Penundaan propagasi konfigurasi siklus hidup

Saat Anda menambahkan konfigurasi Siklus Hidup S3 ke dalam bucket, biasanya ada beberapa jeda sebelum konfigurasi Siklus Hidup baru atau yang diperbarui sepenuhnya disebarkan ke semua sistem Amazon S3. Perkirakan penundaan beberapa menit sebelum konfigurasi benar-benar berpengaruh. Penundaan ini juga dapat terjadi saat Anda menghapus konfigurasi Siklus Hidup S3.

Transisi atau penundaan kedaluwarsa

Ada penundaan antara saat aturan siklus hidup terpenuhi dan saat tindakan untuk aturan selesai. Misalnya, seperangkat objek kedaluwarsa oleh aturan siklus hidup pada 1 Januari. Meskipun aturan kedaluwarsa telah dipenuhi pada 1 Januari, Amazon S3 mungkin tidak benar-benar menghapus objek ini sampai berhari-hari atau bahkan berminggu-minggu kemudian. Penundaan ini terjadi karena Siklus Hidup S3 mengantri objek untuk transisi atau kedaluwarsa secara asinkron. Namun, perubahan dalam penagihan biasanya diterapkan ketika aturan siklus hidup dipenuhi, bahkan jika tindakan tidak selesai. Untuk informasi selengkapnya, lihat Perubahan dalam penagihan. Untuk memantau efek pembaruan yang dibuat oleh aturan siklus hidup aktif, lihat the section called "Bagaimana cara memantau tindakan yang diambil oleh aturan siklus hidup saya?"

Menonaktifkan atau menghapus aturan siklus hidup

Saat Anda menonaktifkan atau menghapus aturan siklus hidup, Amazon S3 berhenti menjadwalkan objek baru untuk dihapus atau dialihkan setelah penundaan kecil. Objek yang sudah dijadwalkan menjadi tidak dijadwalkan dan tidak dihapus atau ditransisikan.

> **Note**
>
> Sebelum memperbarui, menonaktifkan, atau menghapus aturan siklus hidup, gunakan operasi `LIST` API (seperti, ListObjectsV2ListObjectVersions, dan ListMultipartUploads) atau untuk memverifikasi Inventaris Amazon S3 bahwa Amazon S3 telah mentransisikan dan menghapus objek yang memenuhi syarat berdasarkan kasus penggunaan Anda. Jika Anda mengalami masalah dengan memperbarui, menonaktifkan, atau menghapus aturan siklus hidup, lihat. Memecahkan masalah Siklus Hidup S3

Objek yang ada dan baru

Ketika Anda menambahkan konfigurasi Siklus Hidup ke dalam bucket, aturan konfigurasi berlaku untuk objek yang ada dan objek yang Anda tambahkan kemudian. Misalnya, jika Anda menambahkan aturan konfigurasi Siklus Hidup hari ini dengan tindakan kedaluwarsa yang menyebabkan objek dengan awalan tertentu kedaluwarsa 30 hari setelah pembuatan, Amazon S3 akan mengantri untuk menghapus objek yang ada yang berusia lebih dari 30 hari dan yang memiliki awalan yang ditentukan.

Memantau pengaruh aturan siklus hidup

Untuk memantau efek pembaruan yang dibuat oleh aturan siklus hidup aktif, lihat the section called "Bagaimana cara memantau tindakan yang diambil oleh aturan siklus hidup saya?"

Perubahan penagihan

Mungkin ada jeda antara saat aturan konfigurasi Siklus Hidup dipenuhi dan saat tindakan yang dipicu dengan memenuhi aturan diambil. Namun, perubahan penagihan terjadi segera setelah aturan konfigurasi Siklus Hidup dipenuhi, meskipun tindakan belum diambil.

Misalnya, setelah waktu kedaluwarsa objek, Anda tidak dikenakan biaya untuk penyimpanan, meskipun objek tidak segera dihapus. Demikian juga, segera setelah waktu transisi objek berlalu, Anda dikenakan biaya tingkat penyimpanan S3 Glacier Flexible Retrieval, bahkan jika objek tidak segera dialihkan ke kelas penyimpanan S3 Glacier Flexible Retrieval.

Namun, transisi siklus hidup ke kelas penyimpanan S3 Intelligent-Tiering adalah pengecualian. Perubahan penagihan tidak terjadi sampai setelah objek bertransisi ke kelas penyimpanan S3 Intelligent-Tiering.

Aturan ganda atau bertentangan

Jika Anda memiliki beberapa aturan dalam konfigurasi Siklus Hidup S3, objek dapat memenuhi syarat untuk beberapa tindakan Siklus Hidup S3 pada hari yang sama. Dalam kasus tersebut, Amazon S3 mengikuti aturan umum ini:

- Penghapusan permanen lebih diutamakan daripada transisi.
- Transisi lebih diutamakan daripada pembuatan penanda hapus.
- Ketika objek memenuhi syarat untuk transisi S3 Glacier Flexible Retrieval dan S3 Standard-IA (atau S3 One Zone-IA) transisi, Amazon S3 memilih transisi S3 Glacier Flexible Retrieval.

Sebagai contoh, lihat Contoh 5: Filter yang tumpang tindih, tindakan siklus hidup yang bertentangan, dan apa yang dilakukan Amazon S3 dengan bucket tanpa versi.

Menggunakan konsol S3

Anda dapat menentukan aturan siklus hidup untuk semua objek atau subset objek dalam bucket dengan menggunakan awalan bersama (nama objek yang dimulai dengan string umum) atau tag. Dalam aturan siklus hidup, Anda dapat menentukan tindakan khusus untuk versi objek saat ini dan noncurrent. Untuk informasi selengkapnya, lihat berikut ini:

- Mengelola siklus hidup penyimpanan Anda
- Menggunakan Penentuan Versi dalam bucket S3

Untuk membuat aturan siklus hidup

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang ingin Anda buatkan aturan siklus hidupnya.
3. Pilih tab Manajemen, dan pilih Buat aturan siklus hidup.
4. Di Nama aturan siklus hidup, masukkan nama untuk aturan Anda.

   Nama dalam bucket harus unik.
5. Pilih cakupan aturan siklus hidup:

   - Untuk menerapkan aturan siklus hidup ini pada semua objek dengan awalan atau tanda khusus, pilih Batasi cakupan ke awalan atau tanda tertentu.
     - Untuk membatasi cakupan dengan awalan, di Awalan, masukkan awalan.
     - Untuk membatasi cakupan berdasarkan tanda, pilih Tambahkan tanda, dan masukkan kunci dan nilai tanda.

     Untuk informasi selengkapnya tentang awalan nama objek, lihat Membuat nama kunci objek. Untuk informasi selengkapnya tentang tag objek, lihat Mengategorikan penyimpanan Anda menggunakan tag.
   - Untuk menerapkan aturan siklus hidup ini ke semua objek di bucket, pilih Aturan ini berlaku untuk semua objek di bucket, lalu pilih Saya mengakui bahwa aturan ini berlaku untuk semua objek di bucket.
6. Untuk memfilter aturan berdasarkan ukuran objek, Anda dapat memilih Tentukan ukuran objek minimum, Tentukan ukuran objek maksimum, atau kedua opsi.

   - Saat Anda menentukan nilai untuk ukuran objek Minimum atau ukuran objek Maksimum, nilainya harus lebih besar dari 0 byte dan hingga 5 TB. Anda dapat menentukan nilai ini dalam bita, KB, MB, atau GB.
   - Saat Anda menentukan kedua nilai, ukuran objek maksimum harus lebih besar dari ukuran objek minimum.

   > **Note**
   >
   > Ukuran objek Minimum dan filter ukuran objek Maksimum mengecualikan nilai yang ditentukan. Misalnya, jika Anda menyetel filter untuk objek kedaluwarsa yang memiliki

ukuran objek Minimum 128 KB, objek yang persis 128 KB tidak akan kedaluwarsa. Sebaliknya, aturan hanya berlaku untuk objek yang berukuran lebih besar dari 128 KB.

7. Di bawah Tindakan aturan siklus hidup, pilih tindakan yang Anda ingin aturan siklus hidup Anda lakukan:

   - Transisikan versi objek saat ini antar kelas penyimpanan
   - Transisikan versi objek sebelumnya antar kelas penyimpanan
   - Akhiri versi objek saat ini

   > **Note**
   >
   > Untuk bucket yang tidak mengaktifkan Versi S3, versi saat ini yang kedaluwarsa menyebabkan Amazon S3 menghapus objek secara permanen. Untuk informasi selengkapnya, lihat the section called "Tindakan Siklus Hidup dan kondisi Penentuan Versi bucket".

   - Hapus secara permanen versi objek sebelumnya
   - Hapus penanda hapus yang kedaluwarsa atau unggahan multibagian yang belum selesai

   Tergantung pada tindakan yang Anda pilih, opsi yang berbeda akan muncul.

8. Untuk mentransisikan versi objek saat ini antar kelas penyimpanan, di bawah Transisikan objek versi saat ini antar kelas penyimpanan:

   a. Dalam transisi kelas Storage, pilih kelas penyimpanan untuk transisi. Untuk daftar kemungkinan transisi, lihatthe section called "Transisi siklus hidup yang didukung". Anda dapat memilih dari kelas penyimpanan berikut:

      - S3 Standard-IA
      - S3 Intelligent-Tiering
      - S3 One Zone-IA
      - S3 Glacier Flexible Retrieval
      - S3 Glacier Deep Archive

   b. Di Hari setelah pembuatan objek, masukkan jumlah hari setelah pembuatan untuk melakukan transisi objek.

Untuk informasi selengkapnya tentang kelas penyimpanan, lihat Menggunakan kelas penyimpanan Amazon S3. Anda dapat menentukan transisi untuk versi objek saat ini atau sebelumnya atau untuk versi saat ini dan versi sebelumnya. Penentuan Versi memungkinkan Anda untuk menyimpan beberapa versi dari sebuah objek di dalam satu bucket. Untuk informasi selengkapnya tentang penentuan versi, lihat Menggunakan konsol S3.

> **Important**
>
> Saat memilih kelas penyimpanan S3 Glacier Flexible Retrieval atau Glacier Deep Archive, objek akan tetap berada di dalam Amazon S3. Anda tidak dapat mengaksesnya secara langsung melalui layanan Amazon S3 Glacier yang terpisah. Untuk informasi selengkapnya, lihat Transisi objek menggunakan Siklus Hidup Amazon S3.

9. Untuk mentransisikan versi noncurrent dari objek antara kelas penyimpanan, di bawah Transition versi noncurrent objek antara kelas penyimpanan:

   a. Dalam transisi kelas Storage, pilih kelas penyimpanan untuk transisi. Untuk daftar kemungkinan transisi, lihatthe section called “Transisi siklus hidup yang didukung”. Anda dapat memilih dari kelas penyimpanan berikut:

      - S3 Standard-IA
      - S3 Intelligent-Tiering
      - S3 One Zone-IA
      - S3 Glacier Flexible Retrieval
      - S3 Glacier Deep Archive

   b. Dalam Hari setelah objek menjadi tidak aktif, masukkan jumlah hari setelah pembuatan untuk transisi objek.

10. Untuk mengakhiri versi objek saat ini, di bawah Akhiri versi objek saat ini, di Jumlah hari setelah pembuatan objek, masukkan jumlah hari.

> **Important**
>
> Dalam bucket nonversioned, tindakan kedaluwarsa menghasilkan Amazon S3 menghapus objek secara permanen. Untuk informasi lebih lanjut tentang tindakan siklus hidup, lihat Elemen untuk mendeskripsikan tindakan siklus hidup.

11. Untuk menghapus objek versi sebelumnya secara permanen, di bawah Hapus objek versi sebelumnya secara permanen, di Hari setelah objek menjadi versi lama, masukkan jumlah hari. Anda dapat secara opsional menentukan jumlah versi yang lebih baru untuk dipertahankan dengan memasukkan nilai di bawah Jumlah versi yang lebih baru untuk dipertahankan.
12. Di Bawah Hapus penanda hapus yang kedaluwarsa atau unggahan multibagian yang belum selesai, pilih Hapus penanda hapus objek kedaluwarsa dan Hapus unggahan multibagian yang belum selesai. Kemudian, masukkan jumlah hari setelah inisiasi pengunggahan multibagian yang ingin Anda akhiri dan bersihkan unggahan multibagian yang belum selesai.

    Untuk informasi lebih lanjut tentang unggahan multibagian, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.
13. Pilih Buat aturan.

    Jika aturan tidak berisi kesalahan apa pun, Amazon S3 mengaktifkannya, dan Anda dapat melihatnya di tab Manajemen pada Aturan siklus hidup.

Untuk informasi tentang AWS CloudFormation templat dan contoh, lihat Bekerja dengan AWS CloudFormation templat dan AWS::S3::Bucketdi Panduan AWS CloudFormation Pengguna.

Menggunakan AWS CLI

Anda dapat menggunakan AWS CLI perintah berikut untuk mengelola konfigurasi Siklus Hidup S3:

- `put-bucket-lifecycle-configuration`
- `get-bucket-lifecycle-configuration`
- `delete-bucket-lifecycle`

Untuk petunjuk tentang pengaturan AWS CLI, lihatMengembangkan dengan Amazon S3 menggunakan AWS CLI.

Konfigurasi Amazon Siklus Hidup S3 adalah berkas XML. Tetapi ketika Anda menggunakan AWS CLI, Anda tidak dapat menentukan format XMLnya. Anda harus menentukan format JSON sebagai gantinya. Berikut ini adalah contoh konfigurasi siklus hidup XMLdan konfigurasi JSON setara yang dapat Anda tentukan dalam sebuah perintah. AWS CLI

Pertimbangkan contoh konfigurasi Siklus Hidup S3 berikut.

Example Contoh 1

Example

XML

```
<LifecycleConfiguration>
    <Rule>
        <ID>ExampleRule</ID>
        <Filter>
           <Prefix>documents/</Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Transition>
           <Days>365</Days>
           <StorageClass>GLACIER</StorageClass>
        </Transition>
        <Expiration>
             <Days>3650</Days>
        </Expiration>
    </Rule>
</LifecycleConfiguration>
```

JSON

```
{
    "Rules": [
        {
            "Filter": {
                "Prefix": "documents/"
            },
            "Status": "Enabled",
            "Transitions": [
                {
                    "Days": 365,
                    "StorageClass": "GLACIER"
                }
            ],
            "Expiration": {
                "Days": 3650
            },
            "ID": "ExampleRule"
```

```
        }
    ]
        }
```

Example Contoh 2

Example

XML

```
<LifecycleConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <Rule>
        <ID>id-1</ID>
        <Expiration>
            <Days>1</Days>
        </Expiration>
        <Filter>
            <And>
                <Prefix>myprefix</Prefix>
                <Tag>
                    <Key>mytagkey1</Key>
                    <Value>mytagvalue1</Value>
                </Tag>
                <Tag>
                    <Key>mytagkey2</Key>
                    <Value>mytagvalue2</Value>
                </Tag>
            </And>
        </Filter>
        <Status>Enabled</Status>
    </Rule>
</LifecycleConfiguration>
```

JSON

```
{
    "Rules": [
        {
            "ID": "id-1",
            "Filter": {
                "And": {
```

```
                    "Prefix": "myprefix",
                    "Tags": [
                        {
                            "Value": "mytagvalue1",
                            "Key": "mytagkey1"
                        },
                        {
                            "Value": "mytagvalue2",
                            "Key": "mytagkey2"
                        }
                    ]
                }
            },
            "Status": "Enabled",
            "Expiration": {
                "Days": 1
            }
        }
    ]
}
```

Anda dapat menguji `put-bucket-lifecycle-configuration` sebagai berikut.

Untuk menguji konfigurasi

1. Simpan konfigurasi Siklus Hidup JSON dalam sebuah file (misalnya,). *`lifecycle.json`*
2. Jalankan AWS CLI perintah berikut untuk menyetel konfigurasi Siklus Hidup di bucket Anda. Ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
$ aws s3api put-bucket-lifecycle-configuration  \
--bucket DOC-EXAMPLE-BUCKET  \
--lifecycle-configuration file://lifecycle.json
```

3. Untuk memverifikasi, ambil konfigurasi Siklus Hidup S3 dengan menggunakan perintah sebagai `get-bucket-lifecycle-configuration` AWS CLI berikut:

```
$ aws s3api get-bucket-lifecycle-configuration  \
--bucket DOC-EXAMPLE-BUCKET
```

4. Untuk menghapus konfigurasi Siklus Hidup S3, gunakan `delete-bucket-lifecycle` AWS CLI perintah sebagai berikut:

```
aws s3api delete-bucket-lifecycle \
--bucket DOC-EXAMPLE-BUCKET
```

Menggunakan AWS SDK

Java

Anda dapat menggunakan AWS SDK for Java untuk mengelola konfigurasi Siklus Hidup S3 bucket. Untuk informasi selengkapnya tentang mengelola konfigurasi Siklus Hidup S3, lihat Mengelola siklus hidup penyimpanan Anda.

> **Note**
>
> Saat Anda menambahkan konfigurasi Siklus Hidup S3 ke dalam bucket, Amazon S3 menggantikan konfigurasi Siklus Hidup bucket saat ini, jika ada. Untuk memperbarui konfigurasi, Anda mengambilnya, membuat perubahan yang diinginkan, kemudian menambahkan konfigurasi yang direvisi ke bucket.

Contoh berikut menunjukkan cara menggunakan konfigurasi Siklus Hidup bucket AWS SDK for Java untuk menambah, memperbarui, dan menghapus konfigurasi Siklus Hidup. Contoh ini melakukan hal berikut:

- Menambahkan konfigurasi Siklus Hidup ke dalam bucket.
- Membuka kembali konfigurasi Siklus Hidup dan memperbaruinya dengan menambahkan aturan lain.
- Menambahkan konfigurasi Siklus Hidup yang telah dimodifikasi ke dalam bucket. Amazon S3 menggantikan konfigurasi yang ada.
- Mengambil konfigurasi lagi dan memverifikasi bahwa ia memiliki jumlah aturan yang tepat dengan mencetak jumlah aturan.
- Menghapus konfigurasi Siklus Hidup dan memverifikasi bahwa konfigurasi telah dihapus dengan mencoba mengambilnya kembali.

Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.BucketLifecycleConfiguration;
import com.amazonaws.services.s3.model.BucketLifecycleConfiguration.Transition;
import com.amazonaws.services.s3.model.StorageClass;
import com.amazonaws.services.s3.model.Tag;
import com.amazonaws.services.s3.model.lifecycle.LifecycleAndOperator;
import com.amazonaws.services.s3.model.lifecycle.LifecycleFilter;
import com.amazonaws.services.s3.model.lifecycle.LifecyclePrefixPredicate;
import com.amazonaws.services.s3.model.lifecycle.LifecycleTagPredicate;

import java.io.IOException;
import java.util.Arrays;

public class LifecycleConfiguration {

        public static void main(String[] args) throws IOException {
                Regions clientRegion = Regions.DEFAULT_REGION;
                String bucketName = "*** Bucket name ***";

                // Create a rule to archive objects with the "glacierobjects/"
 prefix to Glacier
                // immediately.
                BucketLifecycleConfiguration.Rule rule1 = new
 BucketLifecycleConfiguration.Rule()
                                .withId("Archive immediately rule")
                                .withFilter(new LifecycleFilter(new
 LifecyclePrefixPredicate("glacierobjects/")))
                                .addTransition(new
 Transition().withDays(0).withStorageClass(StorageClass.Glacier))
                                .withStatus(BucketLifecycleConfiguration.ENABLED);

                // Create a rule to transition objects to the Standard-Infrequent
 Access storage
                // class
                // after 30 days, then to Glacier after 365 days. Amazon S3 will
 delete the
                // objects after 3650 days.
```

```
                // The rule applies to all objects with the tag "archive" set to
"true".
                BucketLifecycleConfiguration.Rule rule2 = new
BucketLifecycleConfiguration.Rule()
                                .withId("Archive and then delete rule")
                                .withFilter(new LifecycleFilter(new
LifecycleTagPredicate(new Tag("archive", "true"))))
                                .addTransition(new Transition().withDays(30)

.withStorageClass(StorageClass.StandardInfrequentAccess))
                                .addTransition(new
Transition().withDays(365).withStorageClass(StorageClass.Glacier))
                                .withExpirationInDays(3650)
                                .withStatus(BucketLifecycleConfiguration.ENABLED);

                // Add the rules to a new BucketLifecycleConfiguration.
                BucketLifecycleConfiguration configuration = new
BucketLifecycleConfiguration()
                                .withRules(Arrays.asList(rule1, rule2));

                try {
                        AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                                        .withCredentials(new
ProfileCredentialsProvider())
                                        .withRegion(clientRegion)
                                        .build();

                        // Save the configuration.
                        s3Client.setBucketLifecycleConfiguration(bucketName,
configuration);

                        // Retrieve the configuration.
                        configuration =
s3Client.getBucketLifecycleConfiguration(bucketName);

                        // Add a new rule with both a prefix predicate and a tag
predicate.
                        configuration.getRules().add(new
BucketLifecycleConfiguration.Rule().withId("NewRule")
                                        .withFilter(new LifecycleFilter(new
LifecycleAndOperator(
                                                                Arrays.asList(new
LifecyclePrefixPredicate("YearlyDocuments/"),
```

```
                                                                                new
LifecycleTagPredicate(new Tag(

   "expire_after",

   "ten_years"))))))
                                        .withExpirationInDays(3650)

.withStatus(BucketLifecycleConfiguration.ENABLED));

                       // Save the configuration.
                       s3Client.setBucketLifecycleConfiguration(bucketName,
configuration);

                       // Retrieve the configuration.
                       configuration =
s3Client.getBucketLifecycleConfiguration(bucketName);

                       // Verify that the configuration now has three rules.
                       configuration =
s3Client.getBucketLifecycleConfiguration(bucketName);
                       System.out.println("Expected # of rules = 3; found: " +
configuration.getRules().size());

                       // Delete the configuration.
                       s3Client.deleteBucketLifecycleConfiguration(bucketName);

                       // Verify that the configuration has been deleted by
attempting to retrieve it.
                       configuration =
s3Client.getBucketLifecycleConfiguration(bucketName);
                       String s = (configuration == null) ? "No configuration
found." : "Configuration found.";
                       System.out.println(s);
               } catch (AmazonServiceException e) {
                       // The call was transmitted successfully, but Amazon S3
couldn't process
                       // it, so it returned an error response.
                       e.printStackTrace();
               } catch (SdkClientException e) {
                       // Amazon S3 couldn't be contacted for a response, or the
client
                       // couldn't parse the response from Amazon S3.
                       e.printStackTrace();
```

```
                    }
            }
}
```

.NET

Anda dapat menggunakan AWS SDK for .NET untuk mengelola konfigurasi Siklus Hidup S3 pada bucket. Untuk informasi selengkapnya tentang mengelola konfigurasi Siklus Hidup, lihat Mengelola siklus hidup penyimpanan Anda.

> **Note**
>
> Saat Anda menambahkan konfigurasi Siklus Hidup, Amazon S3 menggantikan konfigurasi yang ada pada bucket yang ditentukan. Untuk memperbarui konfigurasi, Anda harus mengembalikan konfigurasi Siklus Hidup terlebih dahulu, membuat perubahan, kemudian menambahkan konfigurasi Siklus Hidup yang telah direvisi ke bucket.

Contoh berikut menunjukkan cara menggunakan konfigurasi Siklus Hidup bucket AWS SDK for .NET untuk menambah, memperbarui, dan menghapus konfigurasi Siklus Hidup bucket. Contoh kode melakukan hal sebagai berikut:

- Menambahkan konfigurasi Siklus Hidup ke dalam bucket.
- Membuka kembali konfigurasi Siklus Hidup dan memperbaruinya dengan menambahkan aturan lain.
- Menambahkan konfigurasi Siklus Hidup yang telah dimodifikasi ke dalam bucket. Amazon S3 menggantikan konfigurasi Siklus Hidup yang ada.
- Mengambil kembali konfigurasi dan memverifikasinya dengan mencetak jumlah aturan dalam konfigurasi.
- Menghapus konfigurasi Siklus Hidup dan memverifikasi penghapusan.

Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
```

```
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class LifecycleTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;
        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            AddUpdateDeleteLifecycleConfigAsync().Wait();
        }

        private static async Task AddUpdateDeleteLifecycleConfigAsync()
        {
            try
            {
                var lifeCycleConfiguration = new LifecycleConfiguration()
                {
                    Rules = new List<LifecycleRule>
                        {
                            new LifecycleRule
                            {
                                 Id = "Archive immediately rule",
                                 Filter = new LifecycleFilter()
                                 {
                                     LifecycleFilterPredicate = new
 LifecyclePrefixPredicate()
                                     {
                                         Prefix = "glacierobjects/"
                                     }
                                 },
                                 Status = LifecycleRuleStatus.Enabled,
                                 Transitions = new List<LifecycleTransition>
                                 {
                                      new LifecycleTransition
                                      {
                                           Days = 0,
```

```
                                        StorageClass = S3StorageClass.Glacier
                                    }
                                },
                            },
                            new LifecycleRule
                            {
                                 Id = "Archive and then delete rule",
                                  Filter = new LifecycleFilter()
                                 {
                                     LifecycleFilterPredicate = new
LifecyclePrefixPredicate()
                                     {
                                         Prefix = "projectdocs/"
                                     }
                                 },
                                 Status = LifecycleRuleStatus.Enabled,
                                 Transitions = new List<LifecycleTransition>
                                 {
                                      new LifecycleTransition
                                      {
                                           Days = 30,
                                           StorageClass =
S3StorageClass.StandardInfrequentAccess
                                      },
                                      new LifecycleTransition
                                      {
                                        Days = 365,
                                        StorageClass = S3StorageClass.Glacier
                                      }
                                 },
                                 Expiration = new LifecycleRuleExpiration()
                                 {
                                       Days = 3650
                                 }
                            }
                        }
                };

                // Add the configuration to the bucket.
                await AddExampleLifecycleConfigAsync(client,
lifeCycleConfiguration);

                // Retrieve an existing configuration.
                lifeCycleConfiguration = await RetrieveLifecycleConfigAsync(client);
```

```
                // Add a new rule.
                lifeCycleConfiguration.Rules.Add(new LifecycleRule
                {
                    Id = "NewRule",
                    Filter = new LifecycleFilter()
                    {
                        LifecycleFilterPredicate = new LifecyclePrefixPredicate()
                        {
                            Prefix = "YearlyDocuments/"
                        }
                    },
                    Expiration = new LifecycleRuleExpiration()
                    {
                        Days = 3650
                    }
                });

                // Add the configuration to the bucket.
                await AddExampleLifecycleConfigAsync(client,
lifeCycleConfiguration);

                // Verify that there are now three rules.
                lifeCycleConfiguration = await RetrieveLifecycleConfigAsync(client);
                Console.WriteLine("Expected # of rulest=3; found:{0}",
lifeCycleConfiguration.Rules.Count);

                // Delete the configuration.
                await RemoveLifecycleConfigAsync(client);

                // Retrieve a nonexistent configuration.
                lifeCycleConfiguration = await RetrieveLifecycleConfigAsync(client);

            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered ***. Message:'{0}' when writing
an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
```

```
        }

        static async Task AddExampleLifecycleConfigAsync(IAmazonS3 client,
 LifecycleConfiguration configuration)
        {

            PutLifecycleConfigurationRequest request = new
 PutLifecycleConfigurationRequest
            {
                BucketName = bucketName,
                Configuration = configuration
            };
            var response = await client.PutLifecycleConfigurationAsync(request);
        }

        static async Task<LifecycleConfiguration>
 RetrieveLifecycleConfigAsync(IAmazonS3 client)
        {
            GetLifecycleConfigurationRequest request = new
 GetLifecycleConfigurationRequest
            {
                BucketName = bucketName
            };
            var response = await client.GetLifecycleConfigurationAsync(request);
            var configuration = response.Configuration;
            return configuration;
        }

        static async Task RemoveLifecycleConfigAsync(IAmazonS3 client)
        {
            DeleteLifecycleConfigurationRequest request = new
 DeleteLifecycleConfigurationRequest
            {
                BucketName = bucketName
            };
            await client.DeleteLifecycleConfigurationAsync(request);
        }
    }
}
```

Ruby

Anda dapat menggunakan konfigurasi AWS SDK for Ruby untuk mengelola Siklus Hidup S3 pada bucket dengan menggunakan Konfigurasi kelas. AWS::S3::BucketLifecycle Untuk

informasi selengkapnya tentang mengelola konfigurasi siklus hidup, lihat Mengelola siklus hidup penyimpanan Anda.

Penggunaan API REST

Bagian berikut dalam Referensi API Amazon Simple Storage Service menjelaskan API REST yang terkait dengan konfigurasi Siklus Hidup S3.

- PutBucketLifecycleConfiguration
- GetBucketLifecycleConfiguration
- DeleteBucketLifecycle

Pemecahan Masalah Siklus Hidup S3

Untuk masalah umum yang mungkin terjadi saat bekerja dengan Siklus Hidup S3, lihat. the section called “Memecahkan masalah siklus hidup”

Siklus hidup dan konfigurasi bucket lainnya

Selain konfigurasi Siklus Hidup S3, Anda dapat mengaitkan konfigurasi lain dengan bucket Anda. Bagian ini menjelaskan bagaimana konfigurasi Siklus Hidup S3 berkaitan dengan konfigurasi bucket lainnya.

Siklus hidup dan Penentuan Versi

Anda dapat menambahkan konfigurasi Siklus Hidup S3 ke bucket tanpa versi dan bucket dengan dukungan Penentuan Versi. Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.

Bucket yang dengan dukungan Penentuan Versi mempertahankan satu versi objek saat ini, dan nol atau lebih versi objek yang lama. Anda dapat menentukan aturan Siklus Hidup terpisah untuk versi objek saat ini dan yang lama.

Untuk informasi selengkapnya, lihat Elemen konfigurasi Siklus Hidup.

> **Important**
>
> Jika Anda memiliki beberapa aturan dalam konfigurasi Siklus Hidup S3, objek dapat memenuhi syarat untuk beberapa tindakan Siklus Hidup S3 pada hari yang sama. Dalam kasus tersebut, Amazon S3 mengikuti aturan umum ini:
>
> - Penghapusan permanen lebih diutamakan daripada transisi.
> - Transisi lebih diutamakan daripada pembuatan penanda hapus.
> - Saat sebuah objek memenuhi syarat untuk transisi S3 Glacier Flexible Retrieval dan S3 Standard-IA (atau S3 One Zone-IA), Amazon S3 memilih transisi S3 Glacier Flexible Retrieval.
>
> Sebagai contoh, lihat Contoh 5: Filter yang tumpang tindih, tindakan siklus hidup yang bertentangan, dan apa yang dilakukan Amazon S3 dengan bucket tanpa versi.

Konfigurasi Siklus Hidup di bucket yang mengaktifkan MFA

Konfigurasi Siklus Hidup pada bucket dengan autentikasi multi-faktor (MFA) yang diaktifkan tidak didukung.

Siklus hidup dan pencatatan

Tindakan Siklus Hidup Amazon S3 tidak ditangkap oleh AWS CloudTrail pencatatan tingkat objek. CloudTrailmenangkap permintaan API yang dibuat ke titik akhir Amazon S3 eksternal, sedangkan tindakan Siklus Hidup S3 dilakukan menggunakan titik akhir Amazon S3 internal. Log akses server Amazon S3 dapat diaktifkan dalam bucket S3 untuk menangkap tindakan terkait Siklus Hidup S3 seperti transisi objek ke kelas penyimpanan lain dan kedaluwarsa objek yang mengakibatkan penghapusan permanen atau penghapusan logis. Untuk informasi selengkapnya, lihat the section called “Pencatatan akses server”.

Jika Anda telah mengaktifkan log di bucket Anda, log akses server Amazon S3 melaporkan hasil dari operasi berikut.

Log operasi	Deskripsi
`S3.EXPIRE.OBJECT`	Amazon S3 menghapus objek secara permanen karena tindakan kedaluwarsa Siklus Hidup.
`S3.CREATE.DELETEMARKER`	Amazon S3 secara logis menghapus versi saat ini dan menambahkan penanda hapus di bucket berkemampuan versi.
`S3.TRANSITION_SIA.OBJECT`	Amazon S3 mentransisikan objek ke kelas penyimpanan S3 Standard-IA.
`S3.TRANSITION_ZIA.OBJECT`	Amazon S3 mentransisikan objek ke kelas penyimpanan S3 Standard-IA.
`S3.TRANSITION_INT.OBJECT`	Amazon S3 mentransisikan objek ke kelas penyimpanan S3 Intelligent-Tiering.
`S3.TRANSITION_GIR.OBJECT`	Amazon S3 memulai transisi objek ke kelas penyimpanan S3 Glacier Instant Retrieval.
`S3.TRANSITION.OBJECT`	Amazon S3 memulai transisi objek ke kelas penyimpanan S3 Glacier Flexible Retrieval.
`S3.TRANSITION_GDA.OBJECT`	Amazon S3 memulai transisi objek ke kelas penyimpanan S3 Glacier Deep Archive.
`S3.DELETE.UPLOAD`	Amazon S3 membatalkan unggahan multipart yang tidak lengkap.

> **Note**
>
> Catatan log akses server Amazon S3 secara umum diberikan atas dasar upaya terbaik dan tidak dapat digunakan untuk menghitung lengkap semua permintaan Amazon S3.

Pemecahan Masalah Siklus Hidup S3

Untuk informasi selengkapnya tentang pemecahan masalah umum dengan Siklus Hidup S3, lihat Memecahkan masalah Siklus Hidup S3.

Info selengkapnya

- Elemen konfigurasi Siklus Hidup
- Transisi ke kelas penyimpanan S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive (pengarsipan objek)
- Menyetel konfigurasi siklus hidup pada bucket

Mengonfigurasi notifikasi peristiwa Siklus Hidup

Anda dapat mengatur notifikasi peristiwa Amazon S3 untuk menerima pemberitahuan saat Amazon S3 menghapus objek atau mentransikannya ke kelas penyimpanan Amazon S3 lainnya mengikuti aturan Siklus Hidup S3.

Dengan menggunakan jenis `LifecycleExpiration` acara, Anda dapat menerima notifikasi setiap kali Amazon S3 menghapus objek berdasarkan konfigurasi Siklus Hidup S3 Anda. Jenis peristiwa `s3:LifecycleExpiration:Delete` akan memberi tahu Anda saat objek dalam bucket yang tidak berversi dihapus. Ini juga memberi tahu Anda ketika versi objek dihapus secara permanen oleh konfigurasi Siklus Hidup S3. Jenis `s3:LifecycleExpiration:DeleteMarkerCreated` peristiwa akan memberi tahu Anda saat Siklus Hidup S3 membuat penanda hapus saat versi objek saat ini dalam bucket berversi dihapus. Untuk informasi selengkapnya, lihat Menghapus versi objek.

Dengan menggunakan jenis `s3:LifecycleTransition` acara, Anda dapat menerima pemberitahuan saat objek dialihkan dari satu kelas penyimpanan Amazon S3 ke kelas penyimpanan Amazon S3 lainnya dengan konfigurasi Siklus Hidup S3.

Amazon S3 dapat menerbitkan pemberitahuan kejadian ke topik Amazon Simple Notification Service (Amazon SNS), atau fungsi Amazon Simple Queue Service (Amazon SQS), atau fungsi AWS Lambda . Untuk informasi selengkapnya, lihat Notifikasi Peristiwa Amazon S3.

Untuk petunjuk tentang cara mengonfigurasi Notifikasi Peristiwa Amazon S3, lihat Mengaktifkan pemberitahuan peristiwa.

Berikut ini adalah contoh pesan yang dikirimkan Amazon S3 untuk menerbitkan peristiwa `s3:LifecycleExpiration:Delete`. Untuk informasi selengkapnya, lihat Struktur pesan peristiwa.

```
{
   "Records":[
      {
         "eventVersion":"2.3",
         "eventSource":"aws:s3",
         "awsRegion":"us-west-2",
         "eventTime":"1970-01-01T00:00:00.000Z",
         "eventName":"LifecycleExpiration:Delete",
         "userIdentity":{
            "principalId":"s3.amazonaws.com"
         },
         "requestParameters":{
            "sourceIPAddress":"s3.amazonaws.com"
         },
         "responseElements":{
            "x-amz-request-id":"C3D13FE58DE4C810",
            "x-amz-id-2":"FMyUVURIY8/IgAtTv8xRjskZQpcIZ9KG4V5Wp6S7S/
JRWeUWerMUE5JgHvANOjpD"
         },
         "s3":{
            "s3SchemaVersion":"1.0",
            "configurationId":"testConfigRule",
            "bucket":{
               "name":"example-s3-bucket",
               "ownerIdentity":{
                  "principalId":"A3NL1KOZZKExample"
               },
               "arn":"arn:aws:s3:::example-s3-bucket"
            },
            "object":{
               "key":"expiration/delete",
               "sequencer":"0055AED6DCD90281E5",
            }
         }
      }
   ]
}
```

Pesan yang dikirim Amazon S3 untuk mempublikasikan `s3:LifecycleTransition` acara juga menyertakan informasi berikut.

```
"lifecycleEventData":{
    "transitionEventData": {
        "destinationStorageClass": the destination storage class for the object
    }
}
```

Elemen konfigurasi Siklus Hidup

Topik

- Elemen ID
- Elemen status
- Elemen filter
- Elemen untuk mendeskripsikan tindakan siklus hidup

Anda menentukan konfigurasi Siklus Hidup Amazon S3 sebagai XMLyang terdiri dari satu atau beberapa aturan Siklus Hidup.

```
<LifecycleConfiguration>
    <Rule>
         ...
    </Rule>
    <Rule>
         ...
    </Rule>
</LifecycleConfiguration>
```

Setiap aturan terdiri atas hal berikut:

- Metadata aturan yang menyertakan ID aturan, dan status yang menunjukkan apakah aturan diaktifkan atau dinonaktifkan. Jika aturan dinonaktifkan, Amazon S3 tidak akan melakukan tindakan yang ditentukan dalam aturan tersebut.
- Filter yang mengidentifikasi objek yang aturan berlaku. Anda dapat menentukan filter dengan menggunakan ukuran objek, key prefix objek, satu atau beberapa tag objek, atau kombinasi filter.
- Satu atau beberapa tindakan transisi atau kedaluwarsa dengan tanggal atau periode waktu dalam masa pakai objek saat Anda ingin Amazon S3 melakukan tindakan tertentu.

Bagian berikut menjelaskan elemen XML dalam konfigurasi Siklus Hidup S3. Untuk contoh konfigurasi, lihat Contoh konfigurasi Siklus Hidup S3.

Elemen ID

Konfigurasi Siklus Hidup S3 dapat memiliki hingga 1.000 aturan. Batas ini tidak dapat disesuaikan. `<ID>`Elemen secara unik mengidentifikasi aturan. Panjang ID dibatasi hingga 255 karakter.

Elemen status

Nilai `<Status>` elemen dapat berupa `Enabled` atau`Disabled`. Jika aturan dinonaktifkan, Amazon S3 tidak akan melakukan tindakan yang ditentukan dalam aturan tersebut.

Elemen filter

Aturan Siklus Hidup dapat diterapkan ke semua atau subset objek dalam bucket berdasarkan `<Filter>` elemen yang Anda tentukan dalam aturan Siklus Hidup.

Anda dapat memfilter objek berdasarkan awalan kunci, tag objek, atau kombinasi keduanya (dalam hal ini Amazon S3 menggunakan `AND` logis untuk menggabungkan filter). Pertimbangkan contoh berikut:

- Menentukan filter dengan menggunakan awalan kunci — Contoh ini menunjukkan aturan Siklus Hidup S3 yang berlaku untuk subset objek berdasarkan awalan nama kunci (). `logs/` Misalnya, aturan Siklus Hidup berlaku untuk objek`logs/mylog.txt`,`logs/temp1.txt`, dan. `logs/test.txt` Aturan tidak berlaku untuk objek `example.jpg`.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
           <Prefix>logs/</Prefix>
        </Filter>
        transition/expiration actions
         ...
    </Rule>
    ...
</LifecycleConfiguration>
```

Jika Anda ingin menerapkan tindakan siklus hidup ke subset objek berdasarkan awalan nama kunci yang berbeda, tentukan aturan terpisah. Pada setiap aturan, tentukan filter berbasis awalan.

Misalnya, untuk mendeskripsikan tindakan siklus hidup objek dengan awalan kunci `projectA/` dan`projectB/`, Anda menentukan dua aturan sebagai berikut:

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
           <Prefix>projectA/</Prefix>
        </Filter>
        transition/expiration actions
         ...
    </Rule>

    <Rule>
        <Filter>
           <Prefix>projectB/</Prefix>
        </Filter>
        transition/expiration actions
         ...
    </Rule>
</LifecycleConfiguration>
```

Untuk informasi selengkapnya tentang kunci objek, lihat Membuat nama kunci objek.

- Menentukan filter berdasarkan tag objek - Dalam contoh berikut, aturan Siklus Hidup menentukan filter berdasarkan tag (*key*) dan value (). *value* Aturan ini kemudian hanya berlaku untuk subset objek dengan tag tertentu.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
           <Tag>
              <Key>key</Key>
              <Value>value</Value>
           </Tag>
        </Filter>
        transition/expiration actions
        ...
    </Rule>
</LifecycleConfiguration>
```

Anda dapat menentukan filter berdasarkan beberapa tag. Anda harus membungkus tag dalam <And> elemen, seperti yang ditunjukkan pada contoh berikut. Aturan tersebut mengarahkan

Amazon S3 untuk melakukan tindakan siklus hidup pada objek dengan dua tag (dengan kunci dan nilai tag khusus).

```
<LifecycleConfiguration>
    <Rule>
      <Filter>
         <And>
            <Tag>
               <Key>key1</Key>
               <Value>value1</Value>
            </Tag>
            <Tag>
               <Key>key2</Key>
               <Value>value2</Value>
            </Tag>
             ...
          </And>
      </Filter>
      transition/expiration actions
    </Rule>
</Lifecycle>
```

Aturan Siklus Hidup berlaku untuk objek yang memiliki kedua tag yang ditentukan. Amazon S3 menjalankan `AND` logis. Perhatikan hal berikut:

- Setiap tag harus sama persis dengan kunci dan nilai. Jika Anda hanya menentukan `<Key>` elemen dan tidak ada `<Value>` elemen, aturan hanya akan berlaku untuk objek yang cocok dengan kunci tag dan yang tidak memiliki nilai yang ditentukan.
- Aturan ini berlaku untuk subset objek yang memiliki semua tag yang ditentukan dalam aturan. Jika sebuah objek memiliki tag tambahan yang ditentukan, aturan tersebut akan tetap berlaku.

> **Note**
>
> Ketika Anda menentukan beberapa tag di filter, setiap kunci tag harus unik.

- Menentukan filter berdasarkan awalan dan satu atau beberapa tag — Dalam aturan Siklus Hidup, Anda dapat menentukan filter berdasarkan awalan kunci dan satu atau beberapa tag. Sekali lagi, Anda harus membungkus semua elemen filter ini dalam `<And>` elemen, sebagai berikut:

```
<LifecycleConfiguration>
    <Rule>
```

```
        <Filter>
          <And>
             <Prefix>key-prefix</Prefix>
             <Tag>
                <Key>key1</Key>
                <Value>value1</Value>
             </Tag>
             <Tag>
                <Key>key2</Key>
                <Value>value2</Value>
             </Tag>
              ...
          </And>
        </Filter>
        <Status>Enabled</Status>
        transition/expiration actions
    </Rule>
</LifecycleConfiguration>
```

Amazon S3 menggabungkan filter ini dengan menggunakan logika. `AND` Artinya, aturan berlaku untuk subset objek dengan key prefix yang ditentukan dan tag yang ditentukan. Filter hanya dapat memiliki satu awalan, dan nol atau lebih tag.

- Anda dapat menentukan filter kosong, yang dalam hal demikian aturan berlaku untuk semua objek dalam bucket.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
        </Filter>
        <Status>Enabled</Status>
        transition/expiration actions
    </Rule>
</LifecycleConfiguration>
```

- Untuk memfilter aturan berdasarkan ukuran objek, Anda dapat menentukan ukuran minimum (`ObjectSizeGreaterThan`) atau ukuran maksimum (`ObjectSizeLessThan`), atau Anda dapat menentukan rentang ukuran objek.

  Besar objek dalam byte. Ukuran filter maksimum adalah 5 TB. Beberapa kelas penyimpanan memiliki batasan ukuran objek minimum. Untuk informasi selengkapnya, lihat Membandingkan kelas penyimpanan Amazon S3.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
            <ObjectSizeGreaterThan>500</ObjectSizeGreaterThan>
        </Filter>
        <Status>Enabled</Status>
        transition/expiration actions
    </Rule>
</LifecycleConfiguration>
```

> **Note**
>
> `ObjectSizeLessThanFilter` `ObjectSizeGreaterThan` dan mengecualikan nilai yang ditentukan. Misalnya, jika Anda menyetel objek berukuran 128 KB hingga 1024 KB untuk berpindah dari kelas penyimpanan Standar S3 ke kelas penyimpanan IA Standar S3, objek yang persis 1024 KB dan 128 KB tidak akan beralih ke IA Standar S3. Sebagai gantinya, aturan hanya akan berlaku untuk objek yang lebih besar dari 128 KB dan berukuran kurang dari 1024 KB.

Jika Anda menentukan rentang ukuran objek, bilangan bulat `ObjectSizeGreaterThan` harus kurang dari nilai `ObjectSizeLessThan`. Saat menggunakan lebih dari satu filter, Anda harus membungkus filter dalam elemen `<And>`. Contoh berikut menunjukkan bagaimana menentukan objek dalam kisaran antara 500 byte dan 64.000 byte.

```
<LifecycleConfiguration>
    <Rule>
        <Filter>
            <And>
                <Prefix>key-prefix</Prefix>
                <ObjectSizeGreaterThan>500</ObjectSizeGreaterThan>
                <ObjectSizeLessThan>64000</ObjectSizeLessThan>
            </And>
        </Filter>
        <Status>Enabled</Status>
        transition/expiration actions
    </Rule>
</LifecycleConfiguration>
```

Elemen untuk mendeskripsikan tindakan siklus hidup

Anda dapat mengarahkan Amazon S3 untuk melakukan tindakan tertentu dalam masa pakai objek dengan menentukan satu atau beberapa tindakan yang telah ditentukan berikut dalam aturan Siklus Hidup S3. Efek tindakan ini tergantung pada keadaan Penentuan Versi bucket Anda.

- **`Transition`**elemen tindakan - Anda menentukan `Transition` tindakan untuk mengalihkan objek dari satu kelas penyimpanan ke kelas penyimpanan lainnya. Untuk informasi selengkapnya tentang objek transisi, lihat Transisi yang didukung dan kendala terkait. Saat tanggal atau periode waktu tertentu dalam masa pakai objek tercapai, Amazon S3 melakukan transisi.

  Untuk bucket berversi (bucket dengan dukungan Penentuan Versi atau dengan ditangguhkan), tindakan `Transition` berlaku untuk versi objek saat ini. Untuk mengelola versi lama, Amazon S3 menentukan tindakan `NoncurrentVersionTransition` (dijelaskan nanti dalam topik ini).

- **`Expiration`**elemen tindakan — `Expiration` Tindakan akan kedaluwarsa objek yang diidentifikasi dalam aturan dan berlaku untuk objek yang memenuhi syarat di salah satu kelas penyimpanan Amazon S3. Untuk informasi selengkapnya tentang kelas penyimpanan, lihat Menggunakan kelas penyimpanan Amazon S3. Amazon S3 membuat semua objek kedaluwarsa tidak tersedia. Apakah objek secara permanen dihapus tergantung pada status Penentuan Versi bucket.
  - Bucket nonversioned — `Expiration` Tindakan ini menghasilkan Amazon S3 menghapus objek secara permanen.
  - Bucket berversi–Untuk bucket berversi (yaitu, dengan dukungan Penentuan Versi atau dengan Penentuan Versi ditangguhkan), ada beberapa pertimbangan yang memandu cara Amazon S3 menangani tindakan `Expiration`. Untuk bucket dengan dukungan Penentuan Versi atau Penentuan Versi ditangguhkan, berlaku berikut ini:
    - Tindakan `Expiration` hanya berlaku untuk versi saat ini (tidak memiliki dampak pada versi objek lama).
    - Amazon S3 tidak mengambil tindakan apa pun jika ada satu atau beberapa versi objek dan penanda hapus adalah versi saat ini.
    - Jika versi objek saat ini adalah satu-satunya versi objek dan juga merupakan penanda hapus (juga disebut sebagai penanda hapus objek kedaluwarsa, ketika semua versi objek dihapus dan Anda hanya memiliki penanda hapus yang tersisa), Amazon S3 menghapus penanda hapus objek kedaluwarsa. Anda juga dapat menggunakan tindakan kedaluwarsa

guna mengarahkan Amazon S3 untuk menghapus penanda hapus objek yang kedaluwarsa. Sebagai contoh, lihat Contoh 7: Menghapus penanda hapus objek kedaluwarsa.

Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.

Pertimbangkan juga hal berikut saat mengatur Amazon S3 untuk mengelola kedaluwarsa:

- Bucket dengan dukungan Penentuan Versi

  Jika versi objek saat ini bukan penanda hapus, Amazon S3 menambahkan penanda hapus dengan ID versi unik. Ini membuat versi saat ini menjadi versi lama, dan penanda hapus menjadi versi saat ini.

- Bucket dengan Penentuan Versi ditangguhkan

  Dalam bucket yang ditangguhkan versi, tindakan kedaluwarsa menyebabkan Amazon S3 membuat penanda hapus dengan ID versi. `null` Penanda hapus ini menggantikan versi objek apa pun dengan ID versi null dalam hierarki versi, yang secara efektif menghapus objek.

Selain itu, Amazon S3 memberikan tindakan berikut yang dapat Anda gunakan untuk mengelola versi objek lama dalam bucket berversi (yaitu, dengan dukungan Penentuan Versi dan Penentuan Versi ditangguhkan).

- **`NoncurrentVersionTransition`**elemen tindakan - Gunakan tindakan ini untuk menentukan kapan Amazon S3 mentransisikan objek ke kelas penyimpanan yang ditentukan. Anda dapat mendasarkan kedaluwarsa ini pada sejumlah hari tertentu sejak objek menjadi tidak aktif. Selain jumlah hari, Anda juga dapat memberikan jumlah maksimum versi noncurrent untuk dipertahankan (antara 1 dan 100). Nilai ini menentukan berapa banyak versi noncurrent yang lebih baru yang harus ada sebelum Amazon S3 dapat melakukan tindakan terkait pada versi tertentu. Amazon S3 akan mentransisikan versi noncurrent tambahan apa pun di luar nomor yang ditentukan untuk dipertahankan.

  Untuk menentukan jumlah maksimum versi noncurrent, Anda juga harus menyediakan `Filter` elemen. Jika Anda tidak menentukan `Filter` elemen, Amazon S3 menghasilkan `InvalidRequest` kesalahan saat Anda memberikan jumlah maksimum versi noncurrent.

  Untuk informasi selengkapnya tentang objek transisi, lihat Transisi yang didukung dan kendala terkait. Untuk detail tentang cara Amazon S3 menghitung tanggal saat Anda menentukan jumlah hari dalam tindakan `NoncurrentVersionTransition`, lihat Aturan siklus hidup: Berdasarkan usia objek.

- **`NoncurrentVersionExpiration`**elemen tindakan - Gunakan tindakan ini untuk mengarahkan Amazon S3 untuk menghapus versi objek yang tidak aktif secara permanen. Objek yang dihapus ini tidak dapat dipulihkan. Anda dapat mendasarkan kedaluwarsa ini pada sejumlah hari tertentu sejak objek menjadi tidak aktif. Selain jumlah hari, Anda juga dapat memberikan jumlah maksimum versi noncurrent untuk dipertahankan (antara 1 dan 100). Nilai ini menentukan berapa banyak versi lama yang harus ada sebelum Amazon S3 dapat melakukan tindakan terkait pada versi tertentu. Amazon S3 akan secara permanen menghapus versi noncurrent tambahan di luar nomor yang ditentukan untuk disimpan.

  Untuk menentukan jumlah maksimum versi noncurrent, Anda juga harus menyediakan `Filter` elemen. Jika Anda tidak menentukan `Filter` elemen, Amazon S3 menghasilkan `InvalidRequest` kesalahan saat Anda memberikan jumlah maksimum versi noncurrent.

  Penghapusan objek lama yang tertunda ini dapat membantu ketika Anda perlu memperbaiki penghapusan atau penghapusan yang tidak disengaja. Misalnya, Anda dapat mengonfigurasi aturan kedaluwarsa untuk menghapus versi lama lima hari setelah menjadi versi lama. Misalnya, pada 1/1/2014 pukul 10:30 UTC, Anda membuat objek bernama `photo.gif` (versi ID 111111). Pada 1/2/2014 pukul 11:30 UTC, Anda secara tidak sengaja menghapus `photo.gif` (ID versi 111111), yang membuat penanda hapus dengan ID versi baru (seperti ID versi 4857693). Sekarang Anda memiliki waktu lima hari untuk memulihkan versi asli `photo.gif` (ID versi 111111) sebelum penghapusan bersifat permanen. Pada 1/8/2014 pukul 00:00 UTC, aturan Siklus Hidup untuk kedaluwarsa berjalan dan dihapus secara permanen `photo.gif` (ID versi 111111), lima hari setelah menjadi versi noncurrent.

  Untuk detail tentang cara Amazon S3 menghitung tanggal saat Anda menentukan jumlah hari dalam tindakan `NoncurrentVersionExpiration`, lihat Aturan siklus hidup: Berdasarkan usia objek.

  > **Note**
  >
  > Konfigurasi siklus hidup kedaluwarsa objek tidak menghapus unggahan multibagian yang tidak lengkap. Untuk menghapus unggahan multibagian yang tidak lengkap, Anda harus menggunakan tindakan konfigurasi `AbortIncompleteMultipartUpload` Siklus Hidup yang dijelaskan nanti di bagian ini.

Selain tindakan transisi dan kedaluwarsa, Anda dapat menggunakan tindakan konfigurasi Siklus Hidup berikut untuk mengarahkan Amazon S3 agar menghentikan unggahan multibagian yang tidak lengkap atau menghapus penanda penghapusan objek kedaluwarsa:

- **`AbortIncompleteMultipartUpload`**elemen tindakan - Gunakan elemen ini untuk mengatur waktu maksimum (dalam beberapa hari) yang Anda inginkan agar unggahan multipart tetap berlangsung. Jika unggahan multibagian yang berlaku (ditentukan oleh nama kunci yang `prefix` ditentukan dalam aturan Siklus Hidup) tidak berhasil diselesaikan dalam jangka waktu yang telah ditentukan, Amazon S3 menghentikan unggahan multibagian yang tidak lengkap. Untuk informasi selengkapnya, lihat Membatalkan unggahan multibagian.

  > **Note**
  > Anda tidak dapat menentukan tindakan siklus hidup ini dalam aturan yang memiliki filter yang menggunakan tag objek.

- **`ExpiredObjectDeleteMarker`**elemen aksi — Dalam bucket berkemampuan versi, penanda hapus dengan nol versi noncurrent disebut sebagai penanda penghapusan objek kedaluwarsa. Anda dapat menggunakan tindakan siklus hidup ini untuk mengarahkan Amazon S3 untuk menghapus penanda penghapusan objek yang kedaluwarsa. Sebagai contoh, lihat Contoh 7: Menghapus penanda hapus objek kedaluwarsa.

  > **Note**
  > Anda tidak dapat menentukan tindakan siklus hidup ini dalam aturan yang memiliki filter yang menggunakan tag objek.

Cara Amazon S3 menghitung berapa lama sebuah objek menjadi versi lama

Pada bucket dengan dukungan Penentuan Versi, Anda dapat memiliki beberapa versi objek. Selalu ada satu versi saat ini, dan nol atau beberapa versi lama. Setiap kali Anda mengunggah sebuah objek, versi saat ini disimpan sebagai versi lama dan versi baru yang ditambahkan, penerusnya, menjadi versi saat ini. Untuk menentukan jumlah hari saat suatu objek menjadi versi lama, Amazon S3 melihat kapan penerusnya dibuat. Amazon S3 menggunakan jumlah hari sejak penerusnya dibuat sejak jumlah hari suatu objek menjadi versi lama.

Memulihkan versi sebelumnya dari objek saat menggunakan konfigurasi Siklus Hidup S3

Seperti yang dijelaskan dalamMemulihkan versi sebelumnya, Anda dapat menggunakan salah satu dari dua metode berikut untuk mengambil versi objek sebelumnya:

- Metode 1 — Salin versi noncurrent dari objek ke bucket yang sama. Objek yang disalin menjadi versi saat ini dari objek tersebut dan semua versi objek dipertahankan.
- Metode 2 — Hapus versi objek saat ini secara permanen. Ketika Anda menghapus versi objek saat ini, Anda, akibatnya, mengubah versi lama ke versi saat ini dari objek tersebut.

Saat Anda menggunakan aturan konfigurasi Siklus Hidup S3 dengan bucket berkemampuan versi, sebaiknya gunakan Metode 1 sebagai praktik terbaik.
Siklus hidup S3 beroperasi di bawah model yang akhirnya konsisten. Versi saat ini yang Anda hapus secara permanen mungkin tidak hilang sampai perubahan menyebar ke semua sistem Amazon S3. (Oleh karena itu, Amazon S3 mungkin untuk sementara tidak mengetahui penghapusan ini.) Sementara itu, aturan siklus hidup yang Anda konfigurasikan untuk mengakhiri objek lama mungkin secara permanen menghapus objek lama, termasuk objek yang ingin Anda pulihkan. Jadi, menyalin versi lama, seperti yang direkomendasikan dalam Metode 1, adalah alternatif yang lebih aman.

Tindakan Siklus Hidup dan kondisi Penentuan Versi bucket

Aturan siklus hidup: Berdasarkan usia objek

Anda dapat menentukan periode waktu, dalam jumlah hari sejak pembuatan (atau modifikasi) objek, ketika Amazon S3 dapat mengambil tindakan yang ditentukan.

Saat Anda menentukan jumlah hari dalam tindkaan `Transition` dan `Expiration` dalam konfigurasi Siklus Hidup S3, perhatikan hal berikut:

- Nilai yang Anda tentukan adalah jumlah hari sejak pembuatan objek saat tindakan akan terjadi.
- Amazon S3 menghitung waktu dengan menambahkan jumlah hari yang ditentukan dalam aturan ke waktu pembuatan objek dan membulatkan waktu yang dihasilkan ke hari berikutnya pada tengah malam UTC. Misalnya, jika sebuah objek dibuat pada 1/15/2014 pukul 10:30 UTC dan Anda menentukan 3 hari dalam aturan transisi, maka tanggal transisi objek akan dihitung sebagai 1/19/2014 00:00 UTC.

> **Note**
>
> Amazon S3 hanya mempertahankan tanggal modifikasi terakhir untuk setiap objek. Misalnya, konsol Amazon S3 menunjukkan tanggal modifikasi terakhir di panel Properti objek. Ketika Anda awalnya membuat objek baru, tanggal ini mencerminkan tanggal bahwa objek dibuat. Jika Anda mengganti objek, tanggal akan berubah sesuai dengan perubahan tersebut. Oleh karena itu, tanggal pembuatan identik dengan Tanggal modifikasi terakhir.

Saat menentukan jumlah hari pada tindakan `NoncurrentVersionTransition` dan `NoncurrentVersionExpiration` dalam konfigurasi Siklus Hidup, perhatikan hal berikut:

- Nilai yang Anda tentukan adalah jumlah hari sejak versi objek menjadi noncurrent (yaitu, ketika objek ditimpa atau dihapus) bahwa Amazon S3 akan melakukan tindakan pada objek atau objek yang ditentukan.
- Amazon S3 menghitung waktu dengan menambahkan jumlah hari yang ditentukan dalam aturan ke waktu ketika versi penerus baru dari objek dibuat dan membulatkan waktu yang dihasilkan ke hari berikutnya pada tengah malam UTC. Misalnya, di bucket Anda, misalkan Anda memiliki versi objek saat ini yang dibuat pada 1/1/2014 pukul 10:30 UTC. Jika versi baru dari objek yang menggantikan versi saat ini dibuat pada 1/15/2014 pukul 10:30 UTC, dan Anda menentukan 3 hari dalam aturan transisi, tanggal transisi objek dihitung sebagai 1/19/2014 00:00 UTC.

Aturan siklus hidup: Berdasarkan tanggal tertentu

Saat menentukan tindakan dalam aturan Siklus Hidup S3, Anda dapat menentukan tanggal kapan Anda ingin Amazon S3 untuk mengambil tindakan. Ketika tanggal tertentu tiba, Amazon S3 menerapkan tindakan ke semua objek yang memenuhi syarat (berdasarkan kriteria filter).

Jika Anda menentukan tindakan Siklus Hidup S3 dengan tanggal sebelumnya, semua objek yang memenuhi syarat akan segera memenuhi syarat untuk tindakan siklus hidup tersebut.

> **Important**
>
> Tindakan berbasis tanggal bukanlah tindakan satu kali. Amazon S3 terus menerapkan tindakan berbasis tanggal bahkan setelah tanggal berlalu, selama status aturan adalah `Enabled`.
>
> Misalnya, Anda menentukan `Expiration` tindakan berbasis tanggal untuk menghapus semua objek (asumsikan bahwa tidak ada filter yang ditentukan dalam aturan). Pada

tanggal yang ditentukan, Amazon S3 menjadikan mengakhiri semua objek di dalam bucket. Amazon S3 juga terus kedaluwarsa objek baru yang Anda buat di ember. Untuk menghentikan tindakan siklus hidup, Anda harus menghapus tindakan dari aturan siklus hidup, menonaktifkan aturan, atau menghapus aturan dari konfigurasi siklus hidup.

Nilai tanggal harus sesuai dengan format ISO 8601. Waktunya adalah selalu tengah malam UTC.

> **Note**
>
> Anda tidak dapat membuat aturan Siklus Hidup berbasis tanggal menggunakan konsol Amazon S3, tetapi Anda dapat melihat, menonaktifkan, atau menghapus aturan tersebut.

Contoh konfigurasi Siklus Hidup S3

Bagian ini memberikan contoh konfigurasi Siklus Hidup S3. Setiap contoh menunjukkan bagaimana Anda dapat menentukan XML dalam setiap contoh skenario.

Topik

- Contoh 1: Menentukan filter
- Contoh 2: Menonaktifkan aturan Siklus Hidup
- Contoh 3: Mengurangi kelas penyimpanan selama masa pakai objek
- Contoh 4: Menentukan beberapa aturan
- Contoh 5: Filter yang tumpang tindih, tindakan siklus hidup yang bertentangan, dan apa yang dilakukan Amazon S3 dengan bucket tanpa versi
- Contoh 6: Menentukan aturan siklus hidup untuk bucket dengan dukungan Penentuan Versi
- Contoh 7: Menghapus penanda hapus objek kedaluwarsa
- Contoh 8: Konfigurasi siklus hidup untuk membatalkan unggahan multibagian
- Contoh 9: Konfigurasi siklus hidup menggunakan aturan berbasis ukuran

Contoh 1: Menentukan filter

Setiap aturan Siklus Hidup S3 mencakup filter yang dapat Anda gunakan untuk mengidentifikasi bagian objek dalam bucket Anda yang berlaku aturan Siklus Hidup S3. Konfigurasi Siklus Hidup S3 berikut menunjukkan contoh cara menentukan filter.

- Dalam aturan konfigurasi Siklus Hidup S3 ini, filter menentukan awalan kunci (`tax/`). Oleh karena itu, aturan berlaku untuk objek dengan awalan nama kunci `tax/`, seperti `tax/doc1.txt` dan `tax/doc2.txt`.

  Aturan tersebut menetapkan dua tindakan yang mengarahkan Amazon S3 untuk melakukan hal berikut:

  - Transisi objek ke kelas penyimpanan S3 Glacier Flexible Retrieval 365 hari (satu tahun) setelah pembuatan.
  - Menghapus objek (tindakan `Expiration`) 3.650 hari (10 tahun) setelah pembuatan.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Transition and Expiration Rule</ID>
    <Filter>
       <Prefix>tax/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>365</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
    <Expiration>
      <Days>3650</Days>
    </Expiration>
  </Rule>
</LifecycleConfiguration>
```

  Alih-alih menentukan usia objek dalam hal hari setelah pembuatan, Anda dapat menentukan tanggal untuk setiap tindakan. Namun, Anda tidak dapat menggunakan `Date` dan `Days` dalam aturan yang sama.

- Jika Anda ingin aturan Siklus Hidup S3 berlaku pada semua objek dalam bucket, tentukan awalan kosong. Dalam konfigurasi berikut, aturan menentukan tindakan `Transition` yang mengarahkan Amazon S3 untuk melakukan transisi objek ke kelas penyimpanan S3 Glacier Flexible Retrieval 0 hari setelah pembuatan. Aturan ini berarti bahwa objek memenuhi syarat untuk arsip ke S3 Glacier Flexible Retrieval pada tengah malam UTC setelah pembuatan. Untuk informasi selengkapnya tentang batasan siklus hidup, lihat Batasan.

```
<LifecycleConfiguration>
  <Rule>
```

```
    <ID>Archive all object same-day upon creation</ID>
    <Filter>
      <Prefix></Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>0</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
  </Rule>
</LifecycleConfiguration>
```

- Anda dapat menentukan nol atau satu awalan nama kunci dan nol atau lebih tag objek pada suatu filter. Kode contoh berikut ini menerapkan aturan Siklus Hidup S3 pada subset objek dengan awalan kunci `tax/` dan untuk objek yang memiliki dua tag dengan kunci dan nilai spesifik. Saat Anda menentukan lebih dari satu filter, Anda harus menyertakan elemen `<And>` seperti yang ditunjukkan (Amazon S3 menerapkan `AND` logis untuk menggabungkan kondisi filter yang ditentukan).

```
...
<Filter>
   <And>
      <Prefix>tax/</Prefix>
      <Tag>
         <Key>key1</Key>
         <Value>value1</Value>
      </Tag>
      <Tag>
         <Key>key2</Key>
         <Value>value2</Value>
      </Tag>
    </And>
</Filter>
...
```

- Anda dapat memfilter objek berdasarkan tag saja. Misalnya, aturan Siklus Hidup S3 berikut berlaku untuk objek yang memiliki dua label yang ditentukan (tidak menyebutkan awalan apa pun).

```
...
<Filter>
   <And>
```

```
      <Tag>
         <Key>key1</Key>
         <Value>value1</Value>
      </Tag>
      <Tag>
         <Key>key2</Key>
         <Value>value2</Value>
      </Tag>
    </And>
</Filter>
...
```

> **Important**
>
> Jika Anda memiliki beberapa aturan dalam konfigurasi Siklus Hidup S3, objek dapat memenuhi syarat untuk beberapa tindakan Siklus Hidup S3 pada hari yang sama. Dalam kasus tersebut, Amazon S3 mengikuti aturan umum ini:
>
> - Penghapusan permanen lebih diutamakan daripada transisi.
> - Transisi lebih diutamakan daripada pembuatan penanda hapus.
> - Saat objek memenuhi syarat untuk transisi S3 Glacier Flexible Retrieval dan S3 Standard-IA (atau S3 One Zone-IA), Amazon S3 memilih transisi Pengambilan Fleksibel S3 Glacier.
>
> Sebagai contoh, lihat Contoh 5: Filter yang tumpang tindih, tindakan siklus hidup yang bertentangan, dan apa yang dilakukan Amazon S3 dengan bucket tanpa versi.

Contoh 2: Menonaktifkan aturan Siklus Hidup

Anda dapat menonaktifkan sementara aturan Siklus Hidup S3. Konfigurasi Siklus Hidup S3 berikut menetapkan dua aturan:

- Aturan 1 mengarahkan Amazon S3 untuk melakukan transisi objek dengan awalan `logs/` ke kelas penyimpanan S3 Glacier Flexible Retrieval segera setelah pembuatan.
- Aturan 2 mengarahkan Amazon S3 untuk melakukan transisi objek dengan awalan `documents/` ke kelas penyimpanan S3 Glacier Flexible Retrieval segera setelah pembuatan.

Dalam konfigurasi, Aturan 1 diaktifkan dan Aturan 2 dinonaktifkan. Amazon S3 mengabaikan aturan yang dinonaktifkan.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule1</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>0</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
  </Rule>
  <Rule>
    <ID>Rule2</ID>
    <Filter>
      <Prefix>documents/</Prefix>
    </Filter>
    <Status>Disabled</Status>
    <Transition>
      <Days>0</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
  </Rule>
</LifecycleConfiguration>
```

Contoh 3: Mengurangi kelas penyimpanan selama masa pakai objek

Dalam contoh ini, Anda menggunakan konfigurasi Siklus Hidup S3 untuk menurunkan kelas penyimpanan objek selama masa aktifnya. Penurunan tingkat dapat membantu mengurangi biaya penyimpanan. Untuk informasi selengkapnya tentang harga, lihat Harga Amazon S3.

Konfigurasi Siklus Hidup S3 berikut ini menentukan aturan yang berlaku untuk objek dengan awalan nama kunci `logs/`. Aturan menetapkan tindakan berikut ini:

- Dua tindakan transisi:
  - Transisi objek ke kelas penyimpanan S3 Standard-IA 30 hari setelah pembuatan.
  - Transisi objek ke kelas penyimpanan S3 Glacier Flexible Retrieval 90 hari setelah pembuatan.

- Satu tindakan kedaluwarsa yang mengarahkan Amazon S3 untuk menghapus objek setahun setelah pembuatan.

```
<LifecycleConfiguration>
  <Rule>
    <ID>example-id</ID>
    <Filter>
       <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>30</Days>
      <StorageClass>STANDARD_IA</StorageClass>
    </Transition>
    <Transition>
      <Days>90</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
    <Expiration>
      <Days>365</Days>
    </Expiration>
  </Rule>
</LifecycleConfiguration>
```

Note

Anda dapat menggunakan satu aturan untuk menjelaskan semua tindakan Siklus Hidup S3 jika semua tindakan berlaku pada set objek yang sama (diidentifikasi oleh filter). Jika tidak, Anda dapat menambahkan beberapa aturan dengan masing-masing menetapkan filter yang berbeda.

Important

Jika Anda memiliki beberapa aturan dalam konfigurasi Siklus Hidup S3, objek dapat memenuhi syarat untuk beberapa tindakan Siklus Hidup S3 pada hari yang sama. Dalam kasus tersebut, Amazon S3 mengikuti aturan umum ini:

- Penghapusan permanen lebih diutamakan daripada transisi.

- Transisi lebih diutamakan daripada pembuatan penanda hapus.
- Saat objek memenuhi syarat untuk transisi S3 Glacier Flexible Retrieval dan S3 Standard-IA (atau S3 One Zone-IA), Amazon S3 memilih transisi Pengambilan Fleksibel S3 Glacier.

Sebagai contoh, lihat Contoh 5: Filter yang tumpang tindih, tindakan siklus hidup yang bertentangan, dan apa yang dilakukan Amazon S3 dengan bucket tanpa versi.

Contoh 4: Menentukan beberapa aturan

Anda dapat menentukan beberapa aturan jika Anda menginginkan tindakan Siklus Hidup S3 yang berbeda dari objek yang berbeda. Konfigurasi Siklus Hidup S3 berikut memiliki dua aturan:

- Aturan 1 berlaku untuk objek dengan awalan nama kunci `classA/`. Aturan ini mengarahkan Amazon S3 untuk melakukan transisi objek ke kelas penyimpanan S3 Glacier Flexible Retrieval satu tahun setelah pembuatan dan mengakhiri objek ini 10 tahun setelah pembuatan.
- Aturan 2 berlaku untuk objek dengan awalan nama kunci `classB/`. Aturan ini mengarahkan Amazon S3 untuk melakukan transisi objek ke kelas penyimpanan S3 Standard-IA 90 hari setelah pembuatan dan menghapusnya satu tahun setelah pembuatan.

```
<LifecycleConfiguration>
    <Rule>
        <ID>ClassADocRule</ID>
        <Filter>
           <Prefix>classA/</Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Transition>
           <Days>365</Days>
           <StorageClass>GLACIER</StorageClass>
        </Transition>
        <Expiration>
             <Days>3650</Days>
        </Expiration>
    </Rule>
    <Rule>
        <ID>ClassBDocRule</ID>
        <Filter>
```

```
            <Prefix>classB/</Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Transition>
           <Days>90</Days>
           <StorageClass>STANDARD_IA</StorageClass>
        </Transition>
        <Expiration>
             <Days>365</Days>
        </Expiration>
    </Rule>
</LifecycleConfiguration>
```

> **Important**
>
> Jika Anda memiliki beberapa aturan dalam konfigurasi Siklus Hidup S3, objek dapat memenuhi syarat untuk beberapa tindakan Siklus Hidup S3 pada hari yang sama. Dalam kasus tersebut, Amazon S3 mengikuti aturan umum ini:
>
> - Penghapusan permanen lebih diutamakan daripada transisi.
> - Transisi lebih diutamakan daripada pembuatan penanda hapus.
> - Saat objek memenuhi syarat untuk transisi S3 Glacier Flexible Retrieval dan S3 Standard-IA (atau S3 One Zone-IA), Amazon S3 memilih transisi Pengambilan Fleksibel S3 Glacier.
>
> Sebagai contoh, lihat Contoh 5: Filter yang tumpang tindih, tindakan siklus hidup yang bertentangan, dan apa yang dilakukan Amazon S3 dengan bucket tanpa versi.

Contoh 5: Filter yang tumpang tindih, tindakan siklus hidup yang bertentangan, dan apa yang dilakukan Amazon S3 dengan bucket tanpa versi

Anda dapat menentukan konfigurasi Siklus Hidup S3 yang kemudian Anda menentukan tindakan, atau awalan yang tumpang tindih.

Umumnya, Siklus Hidup S3 mengoptimalkan biaya. Misalnya, jika dua kebijakan kedaluwarsa tumpang tindih, kebijakan kedaluwarsa yang lebih pendek akan dipenuhi sehingga data tidak disimpan lebih lama dari yang diharapkan. Demikian pula, jika dua kebijakan transisi tumpang tindih, Siklus Hidup S3 mentransisi objek Anda ke kelas penyimpanan berbiaya rendah.

Dalam kedua kasus tersebut, Siklus Hidup S3 mencoba memilih jalur yang paling murah bagi Anda. Pengecualian untuk aturan umum ini adalah dengan kelas penyimpanan S3 Intelligent-Tiering. S3 Intelligent-Tiering lebih disukai oleh Siklus Hidup S3 daripada kelas penyimpanan mana pun, selain kelas penyimpanan S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive.

Contoh berikut menunjukkan cara Amazon S3 menyelesaikan potensi konflik.

Example 1: Awalan tumpang-tindih (tidak ada konflik)

Konfigurasi contoh berikut memiliki dua aturan yang menetapkan awalan yang tumpang tindih sebagai berikut:

- Aturan pertama menetapkan filter kosong, yang menunjukkan semua objek dalam bucket.
- Aturan kedua menetapkan awalan nama kunci (`logs/`), yang hanya menunjukkan subset objek.

Aturan 1 meminta Amazon S3 menghapus semua objek satu tahun setelah pembuatan. Aturan 2 meminta Amazon S3 mentransisikan subset objek ke kelas penyimpanan S3 Standard-IA 30 hari setelah pembuatan.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule 1</ID>
    <Filter>
    </Filter>
    <Status>Enabled</Status>
    <Expiration>
      <Days>365</Days>
    </Expiration>
  </Rule>
  <Rule>
    <ID>Rule 2</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>STANDARD_IA<StorageClass>
      <Days>30</Days>
    </Transition>
   </Rule>
</LifecycleConfiguration>
```

Karena tidak ada konflik dalam kasus ini, Amazon S3 akan mentransisikan objek dengan awalan `logs/` ke kelas penyimpanan S3 Standard-IA 30 hari setelah pembuatan. Ketika objek apa pun mencapai satu tahun setelah pembuatan, itu akan dihapus.

Example 2: Tindakan siklus hidup yang bertentangan

Dalam konfigurasi contoh ini, ada dua aturan yang mengarahkan Amazon S3 untuk melakukan dua tindakan berbeda pada set objek yang sama dalam masa aktif objek:

- Kedua aturan menentukan awalan nama kunci yang sama, sehingga kedua aturan berlaku pada objek yang sama.
- Kedua aturan menentukan 365 hari yang sama setelah pembuatan objek saat aturan berlaku.
- Satu aturan mengarahkan Amazon S3 untuk mentransisi objek ke kelas penyimpanan S3 Standard-IA dan aturan lain ingin Amazon S3 mengakhiri objek pada saat yang sama.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule 1</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Expiration>
      <Days>365</Days>
    </Expiration>
  </Rule>
  <Rule>
    <ID>Rule 2</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>STANDARD_IA<StorageClass>
      <Days>365</Days>
    </Transition>
   </Rule>
</LifecycleConfiguration>
```

Dalam kasus ini, karena Anda ingin objek kedaluwarsa (dihapus), tidak ada artinya mengubah kelas penyimpanan, jadi Amazon S3 memilih tindakan kedaluwarsa pada objek tersebut.

Example 3: Awalan tumpang-tindih yang mengakibatkan tindakan siklus hidup yang bertentangan

Dalam contoh ini, konfigurasi memiliki dua aturan, yang menetapkan awalan tumpang tindih sebagai berikut:

- Aturan 1 menetapkan awalan kosong (menunjukkan semua objek).
- Aturan 2 menetapkan awalan nama kunci (`logs/`) yang mengidentifikasi bagian dari semua objek.

Untuk subset objek dengan awalan nama kunci `logs/`, tindakan Siklus Hidup S3 dalam kedua aturan berlaku. Satu aturan mengarahkan Amazon S3 transisi objek 10 hari setelah pembuatan, dan aturan lain mengarahkan Amazon S3 mentransisi objek 365 hari setelah pembuatan.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule 1</ID>
    <Filter>
      <Prefix></Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>STANDARD_IA<StorageClass>
      <Days>10</Days>
    </Transition>
  </Rule>
  <Rule>
    <ID>Rule 2</ID>
    <Filter>
      <Prefix>logs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>STANDARD_IA<StorageClass>
      <Days>365</Days>
    </Transition>
   </Rule>
</LifecycleConfiguration>
```

Dalam kasus ini, Amazon S3 memilih untuk melakukan transisi 10 hari setelah pembuatan.

Example 4: Pemfilteran berbasis tag dan tindakan siklus hidup bertentangan yang diakibatkan

Anggaplah Anda memiliki konfigurasi Siklus Hidup S3 berikut yang memiliki dua aturan, masing-masing menetapkan filter tag:

- Aturan 1 menetapkan filter berbasis tag (`tag1/value1`). Aturan ini mengarahkan Amazon S3 mentransisi objek ke kelas penyimpanan S3 Glacier Flexible Retrieval 365 hari setelah pembuatan.
- Aturan 2 menetapkan filter berbasis tag (`tag2/value2`). Aturan ini mengarahkan Amazon S3 untuk mengakhiri objek 14 hari setelah pembuatan.

Konfigurasi Siklus Hidup S3 ditampilkan dalam contoh berikut.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Rule 1</ID>
    <Filter>
      <Tag>
         <Key>tag1</Key>
         <Value>value1</Value>
      </Tag>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <StorageClass>GLACIER<StorageClass>
      <Days>365</Days>
    </Transition>
  </Rule>
  <Rule>
    <ID>Rule 2</ID>
    <Filter>
      <Tag>
         <Key>tag2</Key>
         <Value>value2</Value>
      </Tag>
    </Filter>
    <Status>Enabled</Status>
    <Expiration>
      <Days>14</Days>
    </Expiration>
   </Rule>
</LifecycleConfiguration>
```

Jika suatu objek memiliki kedua tag, Amazon S3 harus memutuskan aturan mana yang harus diikuti. Dalam kasus ini, Amazon S3 mengakhiri objek 14 hari setelah pembuatan. Objek dihapus, dan oleh karena itu tindakan transisi tidak berlaku.

> **Important**
>
> Jika Anda memiliki beberapa aturan dalam konfigurasi Siklus Hidup S3, objek dapat memenuhi syarat untuk beberapa tindakan Siklus Hidup S3 pada hari yang sama. Dalam kasus tersebut, Amazon S3 mengikuti aturan umum ini:
>
> - Penghapusan permanen lebih diutamakan daripada transisi.
> - Transisi lebih diutamakan daripada pembuatan penanda hapus.
> - Saat objek memenuhi syarat untuk transisi S3 Glacier Flexible Retrieval dan S3 Standard-IA (atau S3 One Zone-IA), Amazon S3 memilih transisi Pengambilan Fleksibel S3 Glacier.
>
> Sebagai contoh, lihat Contoh 5: Filter yang tumpang tindih, tindakan siklus hidup yang bertentangan, dan apa yang dilakukan Amazon S3 dengan bucket tanpa versi.

Contoh 6: Menentukan aturan siklus hidup untuk bucket dengan dukungan Penentuan Versi

Misalkan Anda memiliki bucket dengan dukungan Penentuan Versi, yang berarti untuk setiap objek Anda memiliki versi saat ini dan nol atau lebih versi lama. (Untuk informasi selengkapnya tentang Penentuan Versi S3, lihat Menggunakan Penentuan Versi dalam bucket S3.) Dalam contoh ini, Anda ingin mempertahankan nilai historis satu tahun, dan menghapus versi lama. Konfigurasi Siklus Hidup S3 mendukung penyimpanan 1 hingga 100 versi objek apa pun.

Untuk menghemat biaya penyimpanan, Anda ingin memindahkan versi lama ke S3 Glacier Flexible Retrieval 30 hari setelah menjadi versi lama (dengan asumsi objek lama ini adalah data dingin yang tidak memerlukan akses waktu nyata). Selain itu, Anda mengharapkan frekuensi akses versi saat ini berkurang 90 hari setelah pembuatan, jadi Anda dapat memilih untuk memindahkan objek ini ke kelas penyimpanan IA Standar S3.

```
<LifecycleConfiguration>
    <Rule>
        <ID>sample-rule</ID>
```

```
        <Filter>
           <Prefix></Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Transition>
           <Days>90</Days>
           <StorageClass>STANDARD_IA</StorageClass>
        </Transition>
        <NoncurrentVersionTransition>
            <NoncurrentDays>30</NoncurrentDays>
            <StorageClass>GLACIER</StorageClass>
        </NoncurrentVersionTransition>
       <NoncurrentVersionExpiration>
            <NewerNoncurrentVersions>5</NewerNoncurrentVersions>
            <NoncurrentDays>365</NoncurrentDays>
       </NoncurrentVersionExpiration>
    </Rule>
</LifecycleConfiguration>
```

Contoh 7: Menghapus penanda hapus objek kedaluwarsa

Bucket dengan dukungan Penentuan Versi memiliki satu versi saat ini dan nol atau lebih versi lama untuk setiap objek. Ketika Anda menghapus sebuah objek, perhatikan hal berikut ini:

- Jika Anda tidak menentukan ID versi dalam permintaan penghapusan Anda, Amazon S3 menambahkan penanda hapus alih-alih menghapus objek. Versi objek saat ini menjadi versi lama, dan penanda hapus menjadi versi saat ini.
- Jika Anda menentukan ID versi dalam permintaan penghapusan Anda, Amazon S3 menghapus versi objek secara permanen (penanda hapus tidak dibuat).
- Penanda hapus dengan nol versi lama disebut sebagai penanda hapus objek kedaluwarsa.

Contoh ini menunjukkan skenario yang dapat membuat penanda hapus objek kedaluwarsa dalam bucket Anda, dan cara Anda dapat menggunakan konfigurasi Siklus Hidup S3 untuk mengarahkan Amazon S3 untuk menghapus penanda hapus objek kedaluwarsa.

Misalkan Anda menulis konfigurasi Siklus Hidup S3 yang menggunakan `NoncurrentVersionExpiration` tindakan untuk menghapus versi noncurrent 30 hari setelah menjadi noncurrent dan mempertahankan paling banyak 10 versi noncurrent, seperti yang ditunjukkan pada contoh berikut.

```
<LifecycleConfiguration>
    <Rule>
        ...
        <NoncurrentVersionExpiration>
            <NewerNoncurrentVersions>10</NewerNoncurrentVersions>
            <NoncurrentDays>30</NoncurrentDays>
        </NoncurrentVersionExpiration>
    </Rule>
</LifecycleConfiguration>
```

Tindakan `NoncurrentVersionExpiration` tidak berlaku untuk versi objek saat ini. Ini hanya menghapus versi lama.

Untuk versi objek saat ini, Anda memiliki opsi berikut untuk mengelola masa aktifnya, tergantung pada apakah versi objek saat ini mengikuti siklus hidup yang ditetapkan dengan baik:

- Versi objek saat ini mengikuti siklus hidup yang ditentukan dengan baik.

  Dalam kasus ini, Anda dapat menggunakan konfigurasi Siklus Hidup S3 dengan tindakan `Expiration` guna mengarahkan Amazon S3 untuk menghapus versi saat ini, seperti yang ditunjukkan pada contoh berikut.

  ```
  <LifecycleConfiguration>
      <Rule>
          ...
          <Expiration>
             <Days>60</Days>
          </Expiration>
          <NoncurrentVersionExpiration>
              <NewerNoncurrentVersions>10</NewerNoncurrentVersions>
              <NoncurrentDays>30</NoncurrentDays>
          </NoncurrentVersionExpiration>
      </Rule>
  </LifecycleConfiguration>
  ```

  Dalam contoh ini, Amazon S3 menghapus versi saat ini 60 hari setelah dibuat dengan menambahkan penanda hapus untuk setiap versi objek saat ini. Proses ini membuat versi saat ini menjadi lama, dan penanda hapus menjadi versi saat ini. Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.

> **Note**
>
> Anda tidak dapat menentukan tag `Days` dan `ExpiredObjectDeleteMarker` pada aturan yang sama. Saat menentukan tag `Days`, Amazon S3 secara otomatis melakukan pembersihan `ExpiredObjectDeleteMarker` saat penanda hapus sudah cukup tua untuk memenuhi kriteria usia. Untuk membersihkan penanda hapus segera setelah menjadi satu-satunya versi, buat aturan terpisah hanya dengan tag `ExpiredObjectDeleteMarker`.

Tindakan `NoncurrentVersionExpiration` dalam konfigurasi Siklus Hidup S3 yang sama akan menghapus objek lama 30 hari setelah menjadi lama. Dengan demikian, dalam contoh ini, semua versi objek dihapus secara permanen 90 hari setelah pembuatan objek. Meskipun penanda hapus objek kedaluwarsa dibuat selama proses ini, Amazon S3 mendeteksi dan menghapus penanda hapus objek kedaluwarsa untuk Anda.

- Versi objek saat ini tidak memiliki siklus hidup yang didefinisikan dengan baik.

  Dalam kasus ini, Anda dapat menghapus objek secara manual ketika Anda tidak memerlukannya, membuat penanda hapus dengan satu atau lebih versi lama. Jika konfigurasi Siklus Hidup S3 dengan tindakan `NoncurrentVersionExpiration` menghapus semua versi lama, sekarang Anda memiliki penanda hapus objek kedaluwarsa.

  Khusus untuk skenario ini, konfigurasi Siklus Hidup S3 menyediakan tindakan `Expiration` yang dapat Anda gunakan untuk menghapus penanda hapus objek kedaluwarsa.

```
<LifecycleConfiguration>
    <Rule>
       <ID>Rule 1</ID>
        <Filter>
          <Prefix>logs/</Prefix>
        </Filter>
        <Status>Enabled</Status>
        <Expiration>
           <ExpiredObjectDeleteMarker>true</ExpiredObjectDeleteMarker>
        </Expiration>
        <NoncurrentVersionExpiration>
            <NewerNoncurrentVersions>10</NewerNoncurrentVersions>
            <NoncurrentDays>30</NoncurrentDays>
```

```
        </NoncurrentVersionExpiration>
    </Rule>
</LifecycleConfiguration>
```

Dengan mengatur elemen `ExpiredObjectDeleteMarker` ke `true` dalam tindakan `Expiration`, Anda mengarahkan Amazon S3 untuk menghapus penanda hapus objek kedaluwarsa.

> **Note**
>
> Saat Anda menggunakan tindakan Siklus Hidup S3 `ExpiredObjectDeleteMarker`, aturan tidak dapat menentukan filter berbasis tag.

Contoh 8: Konfigurasi siklus hidup untuk membatalkan unggahan multibagian

Anda dapat menggunakan operasi API REST unggahan multibagian Amazon S3 untuk mengunggah objek besar dalam beberapa bagian. Untuk informasi lebih lanjut tentang unggahan multibagian, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.

Dengan menggunakan konfigurasi Siklus Hidup S3, Anda dapat mengarahkan Amazon S3 untuk menghentikan unggahan multibagian yang tidak lengkap (diidentifikasi dengan awalan nama kunci yang ditentukan dalam aturan) jika tidak selesai dalam jumlah hari tertentu setelah inisiasi. Saat Amazon S3 membatalkan unggahan multibagian, Amazon S3 menghapus semua bagian yang terkait dengan unggahan multibagian. Proses ini membantu mengontrol biaya penyimpanan dengan memastikan bahwa Anda tidak memiliki unggahan multibagian yang belum selesai dengan bagian-bagian yang disimpan di Amazon S3.

> **Note**
>
> Saat Anda menggunakan tindakan Siklus Hidup S3 `AbortIncompleteMultipartUpload`, aturan tidak dapat menentukan filter berbasis tag.

Berikut ini adalah contoh konfigurasi Siklus Hidup S3 yang menentukan aturan dengan tinndakan `AbortIncompleteMultipartUpload`. Tindakan ini mengarahkan Amazon S3 untuk menghentikan pengunggahan multibagian yang tidak selesai tujuh hari setelah inisiasi.

```
<LifecycleConfiguration>
    <Rule>
```

```
        <ID>sample-rule</ID>
        <Filter>
           <Prefix>SomeKeyPrefix/</Prefix>
        </Filter>
        <Status>rule-status</Status>
        <AbortIncompleteMultipartUpload>
          <DaysAfterInitiation>7</DaysAfterInitiation>
        </AbortIncompleteMultipartUpload>
    </Rule>
</LifecycleConfiguration>
```

Contoh 9: Konfigurasi siklus hidup menggunakan aturan berbasis ukuran

Anda dapat membuat aturan yang mentransisikan objek hanya berdasarkan ukurannya. Anda dapat menentukan ukuran minimum (`ObjectSizeGreaterThan`) atau ukuran maksimum (`ObjectSizeLessThan`), atau Anda dapat menentukan rentang ukuran objek dalam bita. Saat menggunakan lebih dari satu filter, seperti awalan dan aturan ukuran, Anda harus membungkus filter dalam elemen `<And>`.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Transition with a prefix and based on size</ID>
    <Filter>
       <And>
          <Prefix>tax/</Prefix>
          <ObjectSizeGreaterThan>500</ObjectSizeGreaterThan>
       </And>
    </Filter>
    <Status>Enabled</Status>
    <Transition>
      <Days>365</Days>
      <StorageClass>GLACIER</StorageClass>
    </Transition>
  </Rule>
</LifecycleConfiguration>
```

Jika Anda menentukan rentang dengan menggunakan kedua elemen `ObjectSizeGreaterThan` dan `ObjectSizeLessThan`, ukuran objek maksimum harus lebih besar dari ukuran objek minimum. Saat menggunakan lebih dari satu filter, Anda harus membungkus filter dalam elemen `<And>`. Contoh berikut menunjukkan bagaimana menentukan objek dalam kisaran antara 500 byte dan 64.000 byte. Saat Anda menentukan rentang, `ObjectSizeLessThan`

filter `ObjectSizeGreaterThan` dan mengecualikan nilai yang ditentukan. Untuk informasi selengkapnya, lihat the section called “Elemen filter”.

```
<LifecycleConfiguration>
    <Rule>
        ...
          <And>
             <ObjectSizeGreaterThan>500</ObjectSizeGreaterThan>
             <ObjectSizeLessThan>64000</ObjectSizeLessThan>
          </And>
    </Rule>
</LifecycleConfiguration>
```

Anda juga dapat membuat aturan untuk secara khusus menghapus objek noncurrent yang tidak memiliki data, termasuk objek penanda hapus noncurrent yang dibuat dalam bucket berkemampuan versi. Contoh berikut menggunakan `NoncurrentVersionExpiration` tindakan untuk menghapus versi noncurrent 30 hari setelah menjadi noncurrent dan mempertahankan paling banyak 10 versi noncurrent objek. Ini juga menggunakan `ObjectSizeLessThan` elemen untuk memfilter hanya objek tanpa data.

```
<LifecycleConfiguration>
  <Rule>
    <ID>Expire noncurrent with size less than 1 byte</ID>
    <Filter>
       <ObjectSizeLessThan>1</ObjectSizeLessThan>
    </Filter>
    <Status>Enabled</Status>
    <NoncurrentVersionExpiration>
       <NewerNoncurrentVersions>10</NewerNoncurrentVersions>
       <NoncurrentDays>30</NoncurrentDays>
    </NoncurrentVersionExpiration>
  </Rule>
</LifecycleConfiguration>
```

Inventaris Amazon S3

> **Important**
>
> Amazon S3 sekarang sudah menerapkan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat enkripsi dasar untuk setiap bucket di Amazon S3. Mulai 5

Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan, dan tanpa adanya dampak pada performa. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ Enkripsi default.

Gunakan Inventaris Amazon S3 untuk membantu mengelola penyimpanan. Misalnya, Anda dapat menggunakannya untuk mengaudit dan melaporkan replikasi dan status enkripsi objek untuk kebutuhan bisnis, kepatuhan, dan regulasi. Anda juga dapat menyederhanakan dan mempercepat alur kerja bisnis dan pekerjaan big data dengan menggunakan Inventaris Amazon S3, yang menyediakan alternatif terjadwal untuk operasi API `List` asinkron Amazon S3. Inventaris Amazon S3 tidak menggunakan operasi `List` API untuk mengaudit objek dan tidak memengaruhi tingkat permintaan bucket.

Inventaris Amazon S3 menyediakan nilai yang dipisahkan koma (CSV), Apache baris kolom yang dioptimalkan (ORC) atau file keluaran Apache Parquet yang mencantumkan objek dan metadata terkaitnya setiap hari atau setiap minggu untuk bucket S3 atau objek dengan awalan bersama (yaitu, objek yang memiliki nama yang dimulai dengan string yang sama). Jika Anda memulai inventaris mingguan, laporan akan dibuat setiap hari Minggu (zona waktu UTC) setelah laporan awal. Untuk informasi tentang ketentuan harga inventaris Amazon S3, lihat Harga Amazon S3.

Anda dapat mengonfigurasi beberapa daftar inventaris untuk satu bucket. Saat mengonfigurasi daftar inventaris, Anda dapat menentukan hal-hal berikut:

- Metadata objek yang akan disertakan di inventaris
- Mencantumkan semua versi objek atau hanya versi saat ini
- Tempat penyimpanan output file daftar inventaris
- Membuat inventaris harian atau mingguan
- Mengenkripsi file daftar inventaris atau tidak

Anda dapat melakukan kueri Inventaris Amazon S3 dengan kueri SQL standar dengan menggunakan Amazon Athena, Amazon Redshift Spectrum, dan alat lainnya, seperti Presto, Apache Hive, dan Apache Spark. Untuk informasi selengkapnya tentang penggunaan Athena untuk mengkueri file inventaris, lihat the section called “Membuat kueri inventaris dengan Athena”.

Bucket sumber dan tujuan

Bucket yang daftar objeknya dibuat oleh inventaris disebut bucket sumber. Bucket tempat menyimpan file daftar inventaris disebut bucket tujuan.

Bucket sumber

Inventaris akan mencantumkan objek yang disimpan di dalam bucket sumber. Anda bisa mendapatkan daftar inventaris untuk seluruh bucket, atau memfilter daftar tersebut berdasarkan awalan nama kunci objek.

Bucket sumber:

- Berisi objek yang tercantum di dalam inventaris
- Berisi konfigurasi untuk inventaris

Bucket tujuan

File daftar inventaris Amazon S3 akan ditulis ke bucket tujuan. Untuk mengelompokkan semua file daftar inventaris di lokasi yang sama di dalam bucket tujuan, Anda dapat menentukan awalan tujuan di konfigurasi inventaris.

Bucket tujuan:

- Berisi daftar file inventaris.
- Berisi file manifes yang mencantumkan semua file daftar inventaris yang disimpan di dalam bucket tujuan. Untuk informasi selengkapnya, lihat Manifes inventaris.
- Harus memiliki kebijakan bucket agar dapat memberikan izin ke Amazon S3 untuk memverifikasi kepemilikan bucket dan izin untuk menulis file ke bucket.
- Harus Wilayah AWS sama dengan ember sumber.
- Bisa sama dengan bucket sumber.
- Dapat dimiliki oleh akun yang berbeda Akun AWS dari akun yang memiliki bucket sumber.

Daftar Inventaris Amazon S3

File daftar inventaris berisi daftar objek di dalam bucket sumber dan metadata untuk setiap objek. File daftar inventaris disimpan di dalam bucket tujuan dengan salah satu format berikut:

- Sebagai file CSV yang dikompresi dengan GZIP
- Sebagai file kolom baris (ORC) yang dioptimalkan Apache dan dikompresi dengan ZLIB
- Sebagai file Apache Parquet yang dikompresi dengan Snappy

> **Note**
>
> Objek di laporan Inventaris Amazon S3 tidak dijamin akan diurutkan dengan urutan tertentu.

File daftar inventaris berisi daftar objek di dalam bucket sumber dan metadata untuk setiap objek yang terdaftar:

- Nama bucket – Nama dari bucket yang digunakan untuk menyimpan inventaris.
- Nama kunci – Nama kunci objek (atau kunci) yang mengidentifikasi objek secara unik di dalam bucket. Saat Anda menggunakan format file CSV, nama kuncinya dikodekan dengan URL dan harus didekodekan sebelum dapat digunakan.
- ID Versi – ID versi objek. Saat Anda mengaktifkan versioning di bucket, Amazon S3 akan menetapkan nomor versi untuk objek yang ditambahkan ke dalam bucket. Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3. (Bidang ini tidak disertakan jika daftar tersebut dikonfigurasi hanya untuk versi objek saat ini.)
- IsLatest – Diatur ke `True` jika objek adalah versi objek saat ini. (Bidang ini tidak disertakan jika daftar tersebut dikonfigurasi hanya untuk versi objek saat ini.)
- Delete marker – Diatur ke `True`, jika objeknya adalah delete marker. Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3. (Bidang ini secara otomatis akan ditambahkan ke laporan jika Anda mengonfigurasi laporan untuk menyertakan semua versi objek).
- Ukuran – Ukuran objek dalam byte, tidak termasuk ukuran unggahan multipart yang tidak lengkap, metadata objek, dan delete marker.
- Tanggal terakhir diubah – Tanggal pembuatan objek atau tanggal modifikasi terakhir, mana saja yang terbaru.
- ETag – Tag entitas (ETag) adalah hash dari objek. ETag menunjukkan perubahan hanya pada konten suatu objek, bukan metadata-nya. ETag dapat berupa ringkasan MD5 dari suatu data objek. Ketentuannya tergantung pada bagaimana objek dibuat dan bagaimana objek dienkripsi.
- Kelas penyimpanan – Kelas penyimpanan yang digunakan untuk menyimpan objek. Atur ke `STANDARD`, `REDUCED_REDUNDANCY`, `STANDARD_IA`, `ONEZONE_IA`, `INTELLIGENT_TIERING`,

GLACIER, DEEP_ARCHIVE, OUTPOSTS, GLACIER_IR, or SNOW. Untuk informasi selengkapnya, lihat Menggunakan kelas penyimpanan Amazon S3.

- Flag unggahan multibagian – Atur ke True jika objek diunggah sebagai unggahan multibagian. Untuk informasi selengkapnya, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.
- Status replikasi – Atur ke PENDING, COMPLETED, FAILED, atau REPLICA. Untuk informasi selengkapnya, lihat Mendapatkan informasi status replikasi.
- Status enkripsi — Status enkripsi sisi server, tergantung pada jenis kunci enkripsi yang digunakan —kunci yang dikelola Amazon S3 (SSE-S3), kunci () (SSE-KMS) AWS Key Management Service , atau kunci AWS KMS yang disediakan pelanggan (SSE-C). Atur ke SSE-S3, SSE-C, SSE-KMS, or NOT-SSE. Status NOT-SSE mengindikasikan bahwa objek tersebut tidak dienkripsi dengan enkripsi sisi server. Untuk informasi selengkapnya, lihat Melindungi data dengan enkripsi.
- Kunci Objek S3 bertahan hingga tanggal – Tanggal sampai objek yang terkunci tidak dapat dihapus. Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek S3.
- Mode retensi Kunci Objek S3 – Atur ke Governance atau Compliance untuk objek yang terkunci. Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek S3.
- Status penahanan hukum Kunci Objek S3 – Atur ke On jika penahanan hukum sudah diterapkan pada suatu objek. Jika belum, atur statusnya ke Off. Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek S3.
- Tingkat akses S3 Intelligent-Tiering – Tingkat akses (sering atau jarang) dari suatu objek jika disimpan di kelas penyimpanan S3 Intelligent-Tiering. Atur ke FREQUENT, INFREQUENT, ARCHIVE_INSTANT_ACCESS, ARCHIVE, atau DEEP_ARCHIVE. Untuk informasi selengkapnya, lihat Kelas penyimpanan untuk mengoptimalkan data secara otomatis dengan pola akses yang berubah atau tidak diketahui.
- Status Kunci Bucket S3 – Atur ke ENABLED atau DISABLED. Menunjukkan apakah suatu objek menggunakan Kunci Bucket S3 untuk SSE-KMS. Untuk informasi selengkapnya, lihat Menggunakan Kunci Bucket Amazon S3.
- Algoritma checksum – Menunjukkan algoritma yang digunakan untuk membuat checksum untuk objek.
- Daftar kontrol akses objek — Daftar kontrol akses (ACL) untuk setiap objek yang mendefinisikan Akun AWS atau kelompok mana yang diberikan akses ke objek ini dan jenis akses yang diberikan. Bidang ACL Objek ditentukan dalam format JSON. Laporan Inventaris S3 menyertakan ACL yang terkait dengan objek di bucket sumber Anda, bahkan saat ACL dinonaktifkan untuk bucket. Lihat

informasi yang lebih lengkap di Menggunakan bidang ACL Objek dan Gambaran umum daftar kontrol akses (ACL).

> **Note**
>
> Bidang Object ACL didefinisikan ke dalam format JSON. Laporan inventaris menampilkan nilai untuk bidang ACL Objek sebagai string berkode base64.
> Misalnya, katakanlah Anda memiliki ACL Objek berikut dalam format JSON:
>
> ```
> {
>  "version": "2022-11-10",
>  "status": "AVAILABLE",
>  "grants": [{
>  "canonicalId": "example-canonical-user-ID",
>  "type": "CanonicalUser",
>  "permission": "READ"
>  }]
> }
> ```
>
> Bidang ACL Objek dikodekan dan ditampilkan sebagai string yang dikodekan dengan base64 berikut ini:
>
>
> ```
> eyJ2ZXJzaW9uIjoiMjAyMi0xMS0xMCIsInN0YXR1cyI6IkFWQUlMQUJMRSIsImdyYW50cyI6W3siY2Fub25pY2Fs
> ```
>
>
> Untuk mendapatkan nilai yang terdekripsi dalam JSON untuk bidang ACL Objek, Anda dapat melakukan kueri pada bidang ini di Amazon Athena. Untuk contoh kueri, lihat Menanyakan Inventaris Amazon S3 dengan Amazon Athena.

- Pemilik objek – Pemilik dari objek tersebut.

> **Note**
>
> Saat objek mencapai akhir masa pakai berdasarkan konfigurasi siklus hidupnya, Amazon S3 akan mengantre objek untuk dihapus dan menghapusnya tidak secara bersamaan. Oleh karena itu, penundaan antara tanggal kedaluwarsa dan tanggal saat Amazon S3 menghapus objek mungkin ditemukan. Laporan inventaris mencakup semua objek yang kedaluwarsa tetapi belum dihapus. Untuk informasi selengkapnya tentang tindakan kedaluwarsa di Siklus Hidup S3, lihat Mengakhiri objek.

Anda sebaiknya membuat kebijakan siklus hidup yang menghapus daftar inventaris lama. Untuk informasi selengkapnya, lihat Mengelola siklus hidup penyimpanan Anda.

Izin `s3:PutInventoryConfiguration` memungkinkan pengguna untuk memilih semua bidang metadata yang tercantum sebelumnya untuk setiap objek saat mengonfigurasi daftar inventaris dan menentukan bucket tujuan untuk menyimpan inventaris. Pengguna dengan akses baca ke objek di bucket tujuan dapat mengakses semua bidang metadata objek yang tersedia di daftar inventaris. Untuk membatasi akses ke laporan inventaris, lihat Berikan izin untuk Inventaris S3 dan analitik S3.

Konsistensi inventaris

Semua objek mungkin tidak akan muncul di setiap daftar inventaris. Daftar inventaris akan memberikan konsistensi akhir untuk permintaan (objek baru dan penimpaan) `PUT` dan permintaan `DELETE`. Setiap daftar inventaris untuk bucket berisi cuplikan item bucket. Daftar inventari ini pada akhirnya konsisten (maksudnya, daftar mungkin tidak akan menyertakan objek yang baru ditambahkan atau dihapus).

Untuk memvalidasi status objek sebelum Anda mengambil tindakan untuk objek tertentu, sebaiknya lakukan permintaan API REST `HeadObject` untuk mengambil metadata untuk objek tersebut, atau memeriksa properti objeknya di konsol Amazon S3. Anda juga dapat memeriksa metadata objek dengan AWS CLI atau SDKS. AWS Untuk informasi lebih lanjut, lihat `HeadObject` di dalam Referensi API Layanan Penyimpanan Sederhana Amazon.

Untuk informasi selengkapnya tentang bekerja dengan Inventaris Amazon S3, lihat topik-topik berikut.

Topik

- Mengonfigurasi Inventaris Amazon S3
- Menyiapkan Notifikasi Peristiwa Amazon S3 untuk penyelesaian inventaris
- Menemukan daftar inventaris
- Menanyakan Inventaris Amazon S3 dengan Amazon Athena
- Mengonversi string ID versi kosong di laporan Inventaris Amazon S3 ke string nol
- Menggunakan bidang ACL Objek

Mengonfigurasi Inventaris Amazon S3

Inventaris Amazon S3 menyediakan daftar file datar dari objek dan metadata, sesuai jadwal yang Anda tentukan. Anda dapat menggunakan Inventaris S3 sebagai alternatif terjadwal untuk operasi

API `List` sinkron Amazon S3. Inventaris S3 menyediakan nilai yang dipisahkan koma (CSV), kolom baris yang dioptimalkan Apache (ORC), atau file output Apache Parquet (Parquet) yang mencantumkan objek Anda dan metadata yang sesuai.

Anda dapat mengonfigurasi Inventaris S3 untuk membuat daftar inventaris setiap hari atau setiap minggu untuk bucket S3 atau untuk objek yang memiliki prefiks yang sama (objek yang memiliki nama yang diawali dengan string yang sama). Untuk informasi selengkapnya, lihat Inventaris Amazon S3.

Bagian ini menjelaskan cara mengonfigurasi inventaris, termasuk detail tentang sumber inventaris dan bucket tujuan.

Topik

- Gambaran Umum
- Membuat kebijakan bucket tujuan
- Memberikan izin kepada Amazon S3 untuk menggunakan CMK untuk enkripsi
- Mengonfigurasi inventaris dengan menggunakan konsol S3
- Penggunaan API REST untuk bekerja dengan Inventaris S3

Gambaran Umum

Inventaris Amazon S3 membantu Anda mengelola penyimpanan dengan membuat daftar objek dalam bucket S3 sesuai jadwal yang telah ditentukan. Anda dapat mengonfigurasi beberapa daftar inventaris untuk satu bucket. Daftar inventaris diterbitkan dalam format file CSV, ORC, atau file Parquet di bucket tujuan.

Cara termudah untuk menyiapkan inventaris adalah dengan menggunakan konsol Amazon S3, tetapi Anda juga dapat menggunakan Amazon S3 REST API AWS Command Line Interface ,AWS CLI(), atau SDK. AWS Konsol menjalankan langkah pertama dari prosedur berikut ini untuk Anda: menambahkan kebijakan bucket ke bucket tujuan.

Untuk menyiapkan Inventaris Amazon S3 untuk bucket S3

1. Tambahkan kebijakan bucket untuk bucket tujuan.

   Anda harus membuat kebijakan bucket di bucket tujuan yang memberikan izin ke Amazon S3 untuk menulis objek ke bucket di lokasi yang ditentukan. Untuk contoh kebijakan, lihat Berikan izin untuk Inventaris S3 dan analitik S3.

2. Konfigurasikan inventaris untuk mencantumkan objek dalam bucket sumber dan terbitkan daftar tersebut ke bucket tujuan.

   Ketika mengkonfigurasi daftar inventaris untuk bucket sumber, Anda perlu menentukan bucket tujuan untuk menyimpan daftar tersebut, dan apakah Anda ingin membuat daftar tersebut secara harian atau mingguan. Anda juga dapat mengonfigurasi apakah akan mencantumkan semua versi objek atau hanya versi saat ini dan metadata objek apa yang akan disertakan.

   Beberapa bidang metadata objek dalam konfigurasi laporan Inventaris S3 bersifat opsional, artinya mereka tersedia secara default tetapi dapat dibatasi saat Anda memberikan izin kepada pengguna. `s3:PutInventoryConfiguration` Anda dapat mengontrol apakah pengguna dapat menyertakan bidang metadata opsional ini dalam laporan mereka dengan menggunakan kunci `s3:InventoryAccessibleOptionalFields` kondisi.

   Untuk informasi selengkapnya tentang bidang metadata opsional yang tersedia di Inventaris S3, lihat di Referensi OptionalFieldsAPI Amazon Simple Storage Service. Untuk informasi selengkapnya tentang membatasi akses ke bidang metadata opsional tertentu dalam konfigurasi inventaris, lihat. Kontrol pembuatan konfigurasi laporan Inventaris S3

   Anda dapat menentukan bahwa file daftar inventaris dienkripsi menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3) atau () kunci terkelola pelanggan (SSE-KMS). AWS Key Management Service AWS KMS

   > **Note**
   >
   > The Kunci yang dikelola AWS (`aws/s3`) tidak didukung untuk enkripsi SSE-KMS dengan S3 Inventory.

   Untuk informasi selengkapnya tentang SSE-S3 dan SSE-KMS, lihat Melindungi data dengan enkripsi di sisi klien. Jika Anda berencana menggunakan enkripsi SSE-KMS, lihat Langkah 3.

   - Untuk informasi tentang cara menggunakan konsol untuk mengonfigurasi daftar inventaris, lihat Mengonfigurasi inventaris dengan menggunakan konsol S3.
   - Untuk menggunakan Amazon S3 API untuk mengonfigurasi daftar inventaris, gunakan operasi PutBucketInventoryConfigurationREST API atau yang setara dari AWS CLI atau AWS SDK.

3. Untuk mengenkripsi file daftar inventaris dengan SSE-KMS, berikan izin kepada Amazon S3 untuk menggunakan AWS KMS key.

Anda dapat mengonfigurasi enkripsi untuk file daftar inventaris menggunakan konsol Amazon S3, Amazon S3 REST API AWS CLI, atau SDK. AWS Apa pun cara yang dipilih, Anda harus memberikan izin kepada Amazon S3 untuk menggunakan CMK agar dapat mengenkripsi file inventaris. Berikan izin kepada Amazon S3 dengan memodifikasi kebijakan kunci untuk CMK yang ingin Anda gunakan untuk mengenkripsi file inventaris. Untuk informasi selengkapnya, lihat Memberikan izin kepada Amazon S3 untuk menggunakan CMK untuk enkripsi.

Bucket tujuan yang menyimpan file daftar inventaris dapat dimiliki oleh Akun AWS yang berbeda dari akun yang memiliki bucket sumber. Jika Anda menggunakan enkripsi SSE-KMS untuk operasi lintas akun Inventaris Amazon S3, sebaiknya gunakan ARN kunci KMS yang memenuhi syarat sepenuhnya saat mengonfigurasi inventaris S3. Untuk informasi selengkapnya, lihat Menggunakan enkripsi SSE-KMS untuk operasi lintas akun dan ServerSideEncryptionByDefault di Referensi API Amazon Simple Storage Service.

Membuat kebijakan bucket tujuan

Jika membuat konfigurasi inventaris melalui konsol Amazon S3, Amazon S3 secara otomatis membuat kebijakan bucket pada bucket tujuan yang memberikan izin tulis ke bucket tersebut kepada Amazon S3. Namun, jika Anda membuat konfigurasi inventaris melalui AWS CLI, AWS SDK, atau Amazon S3 REST API, Anda harus menambahkan kebijakan bucket secara manual di bucket tujuan. Untuk informasi selengkapnya, lihat Berikan izin untuk Inventaris S3 dan analitik S3. Kebijakan bucket tujuan Inventaris S3 memungkinkan Amazon S3 menulis data untuk laporan inventaris ke bucket.

Jika terjadi kesalahan saat mencoba membuat kebijakan bucket, Anda akan diberi petunjuk tentang cara memperbaikinya. Misalnya, jika Anda memilih bucket tujuan di bucket lain Akun AWS dan tidak memiliki izin untuk membaca dan menulis ke kebijakan bucket, Anda akan melihat pesan galat.

Dalam hal ini, pemilik bucket tujuan harus menambahkan kebijakan bucket ke bucket tujuan. Jika kebijakan tidak ditambahkan ke bucket tujuan, Anda tidak akan mendapatkan laporan inventaris karena Amazon S3 tidak memiliki izin untuk menulis ke bucket tujuan. Jika bucket sumber dimiliki oleh akun yang berbeda dengan akun pengguna saat ini, ID akun yang benar dari pemilik bucket sumber harus diganti dalam kebijakan.

Memberikan izin kepada Amazon S3 untuk menggunakan CMK untuk enkripsi

Untuk memberikan izin Amazon S3 untuk menggunakan AWS Key Management Service (AWS KMS) kunci terkelola pelanggan untuk enkripsi sisi server, Anda harus menggunakan kebijakan kunci.

Untuk memperbarui kebijakan kunci agar Anda dapat menggunakan CMK, gunakan prosedur berikut ini.

Untuk memberikan izin Amazon S3 untuk mengenkripsi dengan menggunakan kunci terkelola pelanggan

1. Menggunakan Akun AWS yang memiliki kunci yang dikelola pelanggan, masuk ke. AWS Management Console
2. Buka AWS KMS konsol di https://console.aws.amazon.com/kms.
3. Untuk mengubah Wilayah AWS, gunakan pemilih Wilayah di sudut kanan atas halaman.
4. Di panel navigasi kiri, pilih CMK.
5. Di bawah CMK, pilih kunci yang dikelola pelanggan yang ingin Anda gunakan untuk mengenkripsi file inventaris.
6. Di bagian Kebijakan Kunci, pilih Beralih ke tampilan kebijakan.
7. Untuk memperbarui kebijakan kunci, pilih Edit.
8. Pada halaman Edit kebijakan kunci, tambahkan baris berikut ke kebijakan kunci yang ada. Untuk *source-account-id* dan *example-s3-source-bucket*, berikan nilai yang sesuai untuk kasus penggunaan Anda.

```
{
    "Sid": "Allow Amazon S3 use of the customer managed key",
    "Effect": "Allow",
    "Principal": {
        "Service": "s3.amazonaws.com"
    },
    "Action": [
        "kms:GenerateDataKey"
    ],
    "Resource": "*",
    "Condition":{
      "StringEquals":{
         "aws:SourceAccount":"source-account-id"
     },
      "ArnLike":{
        "aws:SourceARN": "arn:aws:s3:::example-s3-source-bucket"
     }
   }
}
```

9. Pilih Simpan perubahan.

Untuk informasi selengkapnya tentang membuat CMK dan menggunakan kebijakan kunci, lihat tautan berikut di Panduan Developer AWS Key Management Service :

- Mengelola kunci
- Kebijakan utama di AWS KMS

Mengonfigurasi inventaris dengan menggunakan konsol S3

Gunakan petunjuk ini untuk mengonfigurasi inventaris dengan menggunakan konsol S3.

> **Note**
>
> Amazon S3 akan membutuhkan waktu hingga 48 jam untuk mengirimkan laporan inventaris pertama.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket. Di daftar Bucket, pilih nama bucket yang ingin Anda konfigurasikan untuk Inventaris Amazon S3.
3. Pilih tab Manajemen.
4. Di bawah Konfigurasi inventaris, pilih Buat konfigurasi inventaris.
5. Untuk Nama konfigurasi inventaris, masukkan nama.
6. Untuk Cakupan inventaris, lakukan hal berikut:
   - Masukkan prefiks opsional.
   - Pilih versi objek yang akan disertakan, baik Versi saat ini saja atau Sertakan semua versi.
7. Di bawah Detail laporan, pilih lokasi Akun AWS tempat Anda ingin menyimpan laporan: Akun ini atau Akun lain.
8. Di bawah Tujuan, pilih bucket tujuan tempat Anda ingin menyimpan laporan inventaris.

   Bucket tujuan harus Wilayah AWS sama dengan ember tempat Anda menyiapkan inventaris. Bucket tujuan dapat berada di Akun AWS yang berbeda. Saat menentukan bucket tujuan, Anda juga dapat menyertakan prefiks opsional untuk mengelompokkan laporan inventaris Anda.

Di bawah bidang bucket Tujuan, terdapat pernyataan Izin bucket tujuan yang ditambahkan ke kebijakan bucket tujuan agar Amazon S3 dapat menempatkan data di bucket tersebut. Untuk informasi selengkapnya, lihat Membuat kebijakan bucket tujuan.

9. Di bawah Frekuensi, pilih seberapa sering laporan akan dibuat: Harian atau Mingguan.
10. Untuk Format output, pilih salah satu format berikut untuk laporan:
    - CSV–Jika Anda berencana untuk menggunakan laporan inventaris ini dengan Operasi Batch S3 atau jika Anda ingin menganalisis laporan ini di alat lain, seperti Microsoft Excel, pilih CSV.
    - Apache ORC
    - Apache Parquet
11. Di bawah Status, pilih Aktifkan atau Nonaktifkan.
12. Untuk mengonfigurasi enkripsi di sisi server, Enkripsi laporan inventaris, ikuti langkah-langkah berikut:

    a. Di bawah Enkripsi sisi server, pilih Jangan tentukan kunci enkripsi atau Tentukan kunci enkripsi untuk mengenkripsi data.

    - Agar pengaturan bucket tetap menggunakan enkripsi di sisi server default untuk objek saat menyimpannya di Amazon S3, pilih Jangan tentukan kunci enkripsi. Selama tujuan bucket telah mengaktifkan Kunci Bucket S3, operasi penyalinan akan menerapkan Kunci Bucket S3 di bucket tujuan.

    > **Note**
    >
    > Jika kebijakan bucket untuk tujuan yang ditentukan mengharuskan objek dienkripsi sebelum menyimpannya di Amazon S3, Anda harus memilih Tentukan kunci enkripsi. Jika tidak, penyalinan objek ke tujuan akan gagal.

    - Untuk mengenkripsi objek sebelum menyimpannya di Amazon S3, pilih Tentukan kunci enkripsi.

    b. Jika Anda memilih Tentukan kunci enkripsi, di bawah Jenis enkripsi, Anda harus memilih kunci terkelola Amazon S3 (SSE-S3) atau AWS Key Management Service kunci (SSE-KMS).

    SSE-S3 menggunakan salah satu penyandian blok terkuat—Advanced Encryption Standard 256-bit (AES-256) untuk mengenkripsi setiap objek. SSE-KMS memberikan kontrol yang

lebih besar terhadap kunci Anda. Untuk informasi selengkapnya tentang SSE-S3, lihat. Menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) Untuk informasi lebih lanjut tentang SSE-KMS, lihat Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS).

> **Note**
>
> Untuk mengenkripsi file daftar inventaris dengan SSE-KMS, Anda harus memberikan izin kepada Amazon S3 untuk menggunakan CMK. Untuk petunjuk, lihat Memberikan Izin kepada Amazon S3 untuk Melakukan Enkripsi Menggunakan Kunci KMS Anda.

c. Jika Anda memilih AWS Key Management Service kunci (SSE-KMS), di bawah AWS KMS key, Anda dapat menentukan AWS KMS kunci Anda melalui salah satu opsi berikut.

> **Note**
>
> Jika bucket tujuan yang menyimpan file daftar inventaris dimiliki oleh yang berbeda Akun AWS, pastikan Anda menggunakan ARN kunci KMS yang memenuhi syarat untuk menentukan kunci KMS Anda.

- Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari AWS KMS kunci Anda, dan pilih kunci KMS enkripsi simetris dari daftar kunci yang tersedia. Pastikan kunci KMS berada di Region yang sama dengan bucket Anda.

  > **Note**
  >
  > Kunci Kunci yang dikelola AWS (`aws/s3`) dan kunci terkelola pelanggan Anda muncul dalam daftar. Namun, Kunci yang dikelola AWS (`aws/s3`) tidak didukung untuk enkripsi SSE-KMS dengan S3 Inventory.

- Untuk memasukkan ARN kunci KMS, pilih Enter AWS KMS key ARN, dan masukkan ARN kunci KMS Anda di bidang yang muncul.
- Untuk membuat kunci terkelola pelanggan baru di AWS KMS konsol, pilih Buat kunci KMS.

13. Untuk Bidang metadata tambahan, pilih satu atau beberapa hal berikut untuk ditambahkan ke laporan inventaris:

    - Ukuran–Ukuran objek dalam byte, tidak termasuk ukuran unggahan multibagian yang tidak lengkap, metadata objek, dan penanda hapus.
    - Tanggal terakhir diubah–Tanggal pembuatan objek atau tanggal modifikasi terakhir, mana pun yang terbaru.
    - Unggahan multibagian–Menentukan bahwa objek diunggah sebagai unggahan multibagian. Untuk informasi selengkapnya, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.
    - Status replikasi–Status replikasi objek. Untuk informasi selengkapnya, lihat Mendapatkan informasi status replikasi.
    - Status enkripsi–Jenis enkripsi di sisi server yang digunakan untuk mengenkripsi objek. Untuk informasi selengkapnya, lihat Melindungi data dengan enkripsi di sisi klien.
    - Status Kunci Bucket - Menunjukkan apakah kunci tingkat ember yang dihasilkan oleh AWS KMS berlaku untuk objek. Untuk informasi selengkapnya, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3.
    - Daftar kontrol akses objek — Daftar kontrol akses (ACL) untuk setiap objek yang mendefinisikan Akun AWS atau kelompok mana yang diberikan akses ke objek ini dan jenis akses yang diberikan. Untuk informasi selengkapnya tentang bidang ini, lihat Menggunakan bidang ACL Objek . Untuk informasi selengkapnya tentang ACL, lihat Gambaran umum daftar kontrol akses (ACL).
    - Pemilik objek–Pemilik objek.
    - Kelas penyimpanan–Kelas penyimpanan yang digunakan untuk menyimpan objek.
    - Intelligent-Tiering: Tingkat Akses–Menunjukkan tingkat akses (sering atau jarang) dari objek jika disimpan di kelas penyimpanan Intelligent-Tiering S3. Untuk informasi selengkapnya, lihat Kelas penyimpanan untuk mengoptimalkan data secara otomatis dengan pola akses yang berubah atau tidak diketahui.
    - ETag–Tag entitas (ETag) adalah hash dari objek. ETag hanya menggambarkan perubahan pada konten objek, bukan pada metadata. ETag mungkin atau mungkin bukan digest MD5 dari data objek. Ketentuannya tergantung pada bagaimana objek dibuat dan bagaimana objek dienkripsi. Untuk informasi selengkapnya, lihat Object dalam Referensi API Amazon Simple Storage Service.
    - Algoritme checksum–Menunjukkan algoritme yang digunakan untuk membuat checksum objek.

- Semua konfigurasi Kunci Objek–Status Kunci Objek, termasuk pengaturan berikut:
  - Kunci Objek: Mode retensi–Tingkat perlindungan yang diterapkan pada objek, baik Tata Kelola atau Kepatuhan.
  - Kunci Objek: Pertahankan hingga tanggal–Tanggal hingga objek yang terkunci tidak dapat dihapus.
  - Kunci Objek: Status penyimpanan yang sah–Status penyimpanan yang sah objek yang terkunci.

  Untuk informasi selngkapnya tentang Kunci Objek S3, lihat Cara kerja Kunci Objek S3.

Untuk informasi selengkapnya tentang konten laporan inventaris, lihat Daftar Inventaris Amazon S3.

Untuk informasi selengkapnya tentang membatasi akses ke bidang metadata opsional tertentu dalam konfigurasi inventaris, lihat. Kontrol pembuatan konfigurasi laporan Inventaris S3

14. Pilih Buat.

Saat daftar inventaris diterbitkan, Anda dapat menanyakan file daftar inventaris dengan Pilihan Amazon S3. Untuk informasi selengkapnya tentang cara menemukan daftar inventaris dan mengkueri file daftar inventaris dengan Pilihan Amazon S3, lihat. Menemukan daftar inventaris

Penggunaan API REST untuk bekerja dengan Inventaris S3

Berikut ini adalah operasi REST yang dapat Anda gunakan untuk bekerja dengan Amazon S3 Inventory.

- DeleteBucketInventoryConfiguration
- GetBucketInventoryConfiguration
- ListBucketInventoryConfigurations
- PutBucketInventoryConfiguration

Menyiapkan Notifikasi Peristiwa Amazon S3 untuk penyelesaian inventaris

Anda dapat mengatur notifikasi peristiwa Amazon S3 untuk menerima pemberitahuan saat file checksum manifes dibuat, yang menandakan bahwa daftar inventaris sudah ditambahkan ke bucket tujuan. Manifes adalah up-to-date daftar semua daftar inventaris di lokasi tujuan.

Amazon S3 dapat menerbitkan peristiwa ke topik Amazon Simple Notification Service (Amazon SNS), antrean Amazon Simple Queue Service (Amazon SQS), atau fungsi AWS Lambda . Untuk informasi selengkapnya, lihat Notifikasi Peristiwa Amazon S3.

Konfigurasi notifikasi berikut menerangkan bahwa semua file `manifest.checksum` yang baru ditambahkan ke bucket tujuan akan diproses oleh AWS Lambda `cloud-function-list-write`.

```
<NotificationConfiguration>
  <QueueConfiguration>
      <Id>1</Id>
      <Filter>
          <S3Key>
              <FilterRule>
                  <Name>prefix</Name>
                  <Value>destination-prefix/source-bucket</Value>
              </FilterRule>
              <FilterRule>
                  <Name>suffix</Name>
                  <Value>checksum</Value>
              </FilterRule>
          </S3Key>
     </Filter>
     <Cloudcode>arn:aws:lambda:us-west-2:222233334444:cloud-function-list-write</
Cloudcode>
     <Event>s3:ObjectCreated:*</Event>
  </QueueConfiguration>
  </NotificationConfiguration>
```

Untuk informasi selengkapnya, lihat Menggunakan AWS Lambda Amazon S3 di Panduan AWS Lambda Pengembang.

Menemukan daftar inventaris

Saat daftar inventaris diterbitkan, file manifes akan diterbitkan ke lokasi berikut di dalam bucket tujuan.

```
destination-prefix/source-bucket/config-ID/YYYY-MM-DDTHH-MMZ/manifest.json
 destination-prefix/source-bucket/config-ID/YYYY-MM-DDTHH-MMZ/manifest.checksum
 destination-prefix/source-bucket/config-ID/hive/dt=YYYY-MM-DD-HH-MM/symlink.txt
```

- *destination-prefix* adalah awalan nama kunci objek yang ditentukan secara opsional didalam konfigurasi inventaris. Gunakan awalan ini untuk mengelompokkan semua file daftar inventaris di lokasi yang sama di dalam bucket tujuan.
- *source-bucket* adalah bucket sumber yang digunakan untuk daftar inventaris. Nama bucket sumber ditambahkan untuk mencegah benturan saat beberapa laporan inventaris dari bucket sumber yang berbeda dikirimkan ke bucket tujuan yang sama.
- *config-ID* ditambahkan untuk mencegah benturan dengan beberapa laporan inventaris dari bucket sumber yang sama yang dikirim ke bucket tujuan yang sama. *config-ID* berasal dari konfigurasi laporan inventaris, dan merupakan nama untuk laporan yang ditentukan selama pengaturan.
- *YYYY-MM-DDTHH-MMZ* adalah stempel waktu yang berisi waktu mulai dan tanggal saat proses pembuatan laporan inventaris memulai pemindaian bucket; misalnya, `2016-11-06T21-32Z`.
- `manifest.json` adalah file manifes.
- `manifest.checksum` adalah hash MD5 dari konten file `manifest.json`.
- `symlink.txt` adalah file manifes yang kompatibel dengan Apache Hive.

Daftar inventaris diterbitkan setiap hari atau setiap minggu ke lokasi berikut di dalam bucket tujuan.

```
destination-prefix/source-bucket/config-ID/data/example-file-name.csv.gz
...
destination-prefix/source-bucket/config-ID/data/example-file-name-1.csv.gz
```

- *destination-prefix* adalah awalan nama kunci objek yang ditentukan secara opsional di dalam konfigurasi inventaris. Gunakan awalan ini untuk mengelompokkan semua file daftar inventaris di lokasi yang sama di dalam bucket tujuan.
- *source-bucket* adalah bucket sumber yang digunakan untuk daftar inventaris. Nama bucket sumber ditambahkan untuk mencegah benturan saat beberapa laporan inventaris dari bucket sumber yang berbeda dikirimkan ke bucket tujuan yang sama.
- *example-file-name*`.csv.gz` adalah salah satu file inventaris CSV. Nama inventaris ORC diakhiri dengan ekstensi nama file `.orc`, dan nama inventaris Parquet diakhiri dengan ekstensi nama file `.parquet`.

  Anda dapat menanyakan tentang file daftar inventaris dengan Amazon S3 Select. *Di konsol Amazon S3, pilih nama daftar inventaris (misalnya, destination-prefix/source-bucket/config-id /data/ .csv.gz). example-file-name* Lalu, pilih

Tindakan objek dan Kueri dengan S3 Select. Untuk contoh cara menggunakan fungsi agregat S3 Select untuk menanyakan file daftar inventaris, lihat Contoh SUM.

Manifes inventaris

File manifes `manifest.json` dan `symlink.txt` menjelaskan lokasi tempat file inventaris berada. Setiap kali ada daftar inventaris baru yang dikirimkan, daftar tersebut akan disertai dengan serangkaian file manifes baru. File-file ini mungkin akan saling menimpa. Di bucket dengan versioning yang diaktifkan, Amazon S3 akan membuat versi baru dari file manifes tersebut.

Setiap manifes yang terdapat di dalam file `manifest.json` menyediakan metadata dan informasi dasar lainnya tentang inventaris. Informasi ini mencakup hal-hal berikut:

- Nama bucket sumber
- Nama bucket tujuan
- Versi inventaris
- Pembuatan stempel waktu dengan format tanggal epoch yang terdiri dari waktu mulai dan tanggal saat proses pembuatan laporan inventaris memulai pemindaian bucket
- Format dan skema file inventaris
- Daftar file inventaris yang berada di dalam bucket tujuan

Setiap kali file `manifest.json` ditulis, file tersebut disertai dengan file `manifest.checksum` yang merupakan hash MD5 dari konten file `manifest.json`.

Example Manifes inventaris di dalam file **`manifest.json`**

Contoh berikut menunjukkan manifes inventaris di dalam file `manifest.json` untuk CSV, ORC, dan inventaris yang diformat Parquet.

CSV

Berikut ini adalah contoh manifes di dalam file `manifest.json` untuk inventaris yang diformat CSV.

```
{
    "sourceBucket": "example-source-bucket",
    "destinationBucket": "arn:aws:s3:::example-inventory-destination-bucket",
    "version": "2016-11-30",
    "creationTimestamp" : "1514944800000",
```

```
    "fileFormat": "CSV",
    "fileSchema": "Bucket, Key, VersionId, IsLatest, IsDeleteMarker,
 Size, LastModifiedDate, ETag, StorageClass, IsMultipartUploaded,
 ReplicationStatus, EncryptionStatus, ObjectLockRetainUntilDate, ObjectLockMode,
 ObjectLockLegalHoldStatus, IntelligentTieringAccessTier, BucketKeyStatus,
 ChecksumAlgorithm, ObjectAccessControlList, ObjectOwner",
    "files": [
        {
            "key": "Inventory/example-source-bucket/2016-11-06T21-32Z/
files/939c6d46-85a9-4ba8-87bd-9db705a579ce.csv.gz",
            "size": 2147483647,
            "MD5checksum": "f11166069f1990abeb9c97ace9cdfabc"
        }
    ]
}
```

ORC

Berikut ini adalah contoh manifes di dalam file `manifest.json` untuk inventaris yang diformat ORC.

```
{
    "sourceBucket": "example-source-bucket",
    "destinationBucket": "arn:aws:s3:::example-destination-bucket",
    "version": "2016-11-30",
    "creationTimestamp" : "1514944800000",
    "fileFormat": "ORC",
    "fileSchema":
 "struct<bucket:string,key:string,version_id:string,is_latest:boolean,is_delete_marker:boole
    "files": [
        {
            "key": "inventory/example-source-bucket/data/
d794c570-95bb-4271-9128-26023c8b4900.orc",
            "size": 56291,
            "MD5checksum": "5925f4e78e1695c2d020b9f6eexample"
        }
    ]
}
```

Parquet

Berikut ini adalah contoh manifes di dalam file `manifest.json` untuk inventaris yang diformat Parquet.

```
{
    "sourceBucket": "example-source-bucket",
    "destinationBucket": "arn:aws:s3:::example-destination-bucket",
    "version": "2016-11-30",
    "creationTimestamp" : "1514944800000",
    "fileFormat": "Parquet",
    "fileSchema": "message s3.inventory { required binary bucket (UTF8);
 required binary key (UTF8); optional binary version_id (UTF8); optional boolean
 is_latest; optional boolean is_delete_marker; optional int64 size; optional
 int64 last_modified_date (TIMESTAMP_MILLIS); optional binary e_tag (UTF8);
 optional binary storage_class (UTF8); optional boolean is_multipart_uploaded;
 optional binary replication_status (UTF8); optional binary encryption_status
 (UTF8); optional int64 object_lock_retain_until_date (TIMESTAMP_MILLIS); optional
 binary object_lock_mode (UTF8); optional binary object_lock_legal_hold_status
 (UTF8); optional binary intelligent_tiering_access_tier (UTF8); optional binary
 bucket_key_status (UTF8); optional binary checksum_algorithm (UTF8); optional
 binary object_access_control_list (UTF8); optional binary object_owner (UTF8);}",
    "files": [
        {
           "key": "inventory/example-source-bucket/data/
d754c470-85bb-4255-9218-47023c8b4910.parquet",
            "size": 56291,
            "MD5checksum": "5825f2e18e1695c2d030b9f6eexample"
        }
    ]
}
```

File `symlink.txt` adalah file manifes yang kompatibel dengan Apache Hive dan memungkinkan Hive untuk menemukan file inventaris dan file data terkait secara otomatis. Manifes yang kompatibel dengan Hive dapat berfungsi untuk layanan yang kompatibel dengan Hive Athena dan Amazon Redshift Spectrum. File ini juga berfungsi di aplikasi yang kompatibel dengan Hive, termasuk Presto, Apache Hive, Apache Spark, dan banyak lainnya.

> **Important**
>
> Saat ini, file manifes `symlink.txt` yang kompatibel dengan Apache Hive tidak berfungsi dengan AWS Glue.
>
> Pembacaan file `symlink.txt` dengan Apache Hive dan Apache Spark tidak didukung untuk ORC dan file inventaris yang diformat dengan Parquet.

Menanyakan Inventaris Amazon S3 dengan Amazon Athena

Anda dapat menanyakan file Inventaris Amazon S3 dengan kueri SQL standar dengan menggunakan Amazon Athena di semua Wilayah tempat Athena tersedia. Untuk memeriksa ketersediaan Wilayah AWS, lihat Tabel Wilayah AWS.

Athena dapat menanyakan file Inventaris Amazon S3 diApachekolom baris yang dioptimalkan (ORC),Apache Parquet, atau format nilai yang dipisahkan koma (CSV). Saat Anda menggunakan Athena untuk menanyakan file inventaris, kami sarankan Anda menggunakan format ORC atauParquet-file inventaris yang diformat. Orc danParquetformat memberikan kinerja kueri yang lebih cepat dan biaya kueri yang lebih rendah. ORC danParquetadalah format file kolumnar yang mendeskripsikan diri dan sadar tipe yang dirancang untukApache Hadoop. Format kolom memungkinkan pembaca untuk membaca, mendekompresi, dan hanya memproses kolom yang diperlukan untuk kueri saat ini. Orc danParquetformat untuk Amazon S3 Inventory tersedia di semuaWilayah AWS.

Untuk menggunakan Athena untuk menanyakan file Inventaris Amazon S3

1. Buat tabel Athena. Untuk informasi cara membuat tabel, lihat Membuat Tabel di Amazon Athena dalam Panduan Pengguna Amazon Athena.
2. Buat kueri Anda dengan menggunakan salah satu contoh templat kueri berikut, tergantung pada apakah Anda menanyakan laporan inventaris berformat ORC, berformat Parket, atau berformat CSV.

   - Saat Anda menggunakan Athena untuk menanyakan laporan inventaris berformat ORC, gunakan contoh kueri berikut sebagai templat.

     Contoh kueri berikut mencakup semua bidang opsional dalam laporan inventaris berformat ORC.

     Untuk menggunakan kueri contoh ini, lakukan hal berikut:

     - Ganti*your_table_name*dengan nama tabel Athena yang Anda buat.
     - Hapus semua bidang opsional yang tidak Anda pilih untuk inventaris Anda sehingga kueri sesuai dengan bidang yang dipilih untuk inventaris Anda.
     - Ganti nama bucket dan lokasi inventaris berikut (ID konfigurasi) yang sesuai untuk konfigurasi Anda.

       ```
       s3://DOC-EXAMPLE-BUCKET/config-ID/hive/
       ```

- Ganti*2022-01-01-00-00*tanggal di bawah`projection.dt.range`dengan hari pertama rentang waktu di mana Anda mempartisi data di Athena. Untuk informasi lebih lanjut, lihat[Mempartisi data di Athena](#).

```
CREATE EXTERNAL TABLE your_table_name(
         bucket string,
         key string,
         version_id string,
         is_latest boolean,
         is_delete_marker boolean,
         size bigint,
         last_modified_date timestamp,
         e_tag string,
         storage_class string,
         is_multipart_uploaded boolean,
         replication_status string,
         encryption_status string,
         object_lock_retain_until_date bigint,
         object_lock_mode string,
         object_lock_legal_hold_status string,
         intelligent_tiering_access_tier string,
         bucket_key_status string,
         checksum_algorithm string,
         object_access_control_list string,
         object_owner string
) PARTITIONED BY (
        dt string
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.orc.OrcSerde'
  STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
  OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.IgnoreKeyTextOutputFormat'
  LOCATION 's3://source-bucket/config-ID/hive/'
  TBLPROPERTIES (
    "projection.enabled" = "true",
    "projection.dt.type" = "date",
    "projection.dt.format" = "yyyy-MM-dd-HH-mm",
    "projection.dt.range" = "2022-01-01-00-00,NOW",
    "projection.dt.interval" = "1",
    "projection.dt.interval.unit" = "HOURS"
  );
```

- Saat Anda menggunakan Athena untuk menanyakanParquet-laporan inventaris yang diformat, gunakan kueri sampel untuk laporan berformat ORC. Namun, gunakan yang berikut iniParquet SerDedi tempat ORCSerDediROW FORMAT SERDEpernyataan.

```
ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
```

- Saat Anda menggunakan Athena untuk menanyakan laporan inventaris berformat CSV, gunakan contoh kueri berikut sebagai templat.

  Contoh kueri berikut mencakup semua bidang opsional dalam laporan inventaris berformat CSV.

  Untuk menggunakan kueri contoh ini, lakukan hal berikut:

  - Ganti*your_table_name*dengan nama tabel Athena yang Anda buat.
  - Hapus semua bidang opsional yang tidak Anda pilih untuk inventaris Anda sehingga kueri sesuai dengan bidang yang dipilih untuk inventaris Anda.
  - Ganti nama bucket dan lokasi inventaris berikut (ID konfigurasi) yang sesuai untuk konfigurasi Anda.

    s3://*DOC-EXAMPLE-BUCKET*/*config-ID*/hive/

  - Ganti*2022-01-01-00-00*tanggal di bawah`projection.dt.range`dengan hari pertama rentang waktu di mana Anda mempartisi data di Athena. Untuk informasi lebih lanjut, lihatMempartisi data di Athena.

```
CREATE EXTERNAL TABLE your_table_name(
         bucket string,
         key string,
         version_id string,
         is_latest boolean,
         is_delete_marker boolean,
         size string,
         last_modified_date string,
         e_tag string,
         storage_class string,
         is_multipart_uploaded boolean,
         replication_status string,
         encryption_status string,
         object_lock_retain_until_date string,
         object_lock_mode string,
         object_lock_legal_hold_status string,
```

```
         intelligent_tiering_access_tier string,
         bucket_key_status string,
         checksum_algorithm string,
         object_access_control_list string,
         object_owner string
) PARTITIONED BY (
        dt string
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
  STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
  OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.IgnoreKeyTextOutputFormat'
  LOCATION 's3://source-bucket/config-ID/hive/'
  TBLPROPERTIES (
    "projection.enabled" = "true",
    "projection.dt.type" = "date",
    "projection.dt.format" = "yyyy-MM-dd-HH-mm",
    "projection.dt.range" = "2022-01-01-00-00,NOW",
    "projection.dt.interval" = "1",
    "projection.dt.interval.unit" = "HOURS"
  );
```

3. Anda sekarang dapat menjalankan berbagai kueri pada inventaris Anda, seperti yang ditunjukkan pada contoh berikut. Ganti masing-masing*user input placeholder*dengan informasi Anda sendiri.

```
# Get a list of the latest inventory report dates available.
SELECT DISTINCT dt FROM your_table_name ORDER BY 1 DESC limit 10;

# Get the encryption status for a provided report date.
SELECT encryption_status, count(*) FROM your_table_name WHERE dt = 'YYYY-MM-DD-HH-
MM' GROUP BY encryption_status;

# Get the encryption status for inventory report dates in the provided range.
SELECT dt, encryption_status, count(*) FROM your_table_name
WHERE dt > 'YYYY-MM-DD-HH-MM' AND dt < 'YYYY-MM-DD-HH-MM' GROUP BY dt,
 encryption_status;
```

Saat Anda mengonfigurasi Inventaris S3 untuk menambahkan bidang Object Access Control List (Object ACL) ke laporan inventaris, laporan akan menampilkan nilai untuk bidang Object ACL sebagai string yang dikodekan base64. Untuk mendapatkan nilai decoded di JSON untuk bidang Object ACL, Anda dapat menanyakan bidang ini dengan menggunakan Athena. Lihat contoh

kueri berikut. Untuk informasi selengkapnya tentang bidang Object ACL, lihat[Menggunakan bidang ACL Objek].

```
# Get the S3 keys that have Object ACL grants with public access.
WITH grants AS (
    SELECT key,
        CAST(
            json_extract(from_utf8(from_base64(object_access_control_list)),
 '$.grants') AS ARRAY(MAP(VARCHAR, VARCHAR))
        ) AS grants_array
    FROM your_table_name
)
SELECT key,
       grants_array,
       grant
FROM grants, UNNEST(grants_array) AS t(grant)
WHERE element_at(grant, 'uri') = 'http://acs.amazonaws.com/groups/global/AllUsers'
```

```
# Get the S3 keys that have Object ACL grantees in addition to the object owner.
WITH grants AS
    (SELECT key,
    from_utf8(from_base64(object_access_control_list)) AS
 object_access_control_list,
         object_owner,
         CAST(json_extract(from_utf8(from_base64(object_access_control_list)),
         '$.grants') AS ARRAY(MAP(VARCHAR, VARCHAR))) AS grants_array
    FROM your_table_name)
SELECT key,
       grant,
       objectowner
FROM grants, UNNEST(grants_array) AS t(grant)
WHERE cardinality(grants_array) > 1 AND element_at(grant, 'canonicalId') !=
 object_owner;
```

```
# Get the S3 keys with READ permission that is granted in the Object ACL.
WITH grants AS (
    SELECT key,
        CAST(
            json_extract(from_utf8(from_base64(object_access_control_list)),
 '$.grants') AS ARRAY(MAP(VARCHAR, VARCHAR))
```

```
        ) AS grants_array
    FROM your_table_name
)
SELECT key,
       grants_array,
       grant
FROM grants, UNNEST(grants_array) AS t(grant)
WHERE element_at(grant, 'permission') = 'READ';
```

```
# Get the S3 keys that have Object ACL grants to a specific canonical user ID.
WITH grants AS (
    SELECT key,
        CAST(
            json_extract(from_utf8(from_base64(object_access_control_list)),
 '$.grants') AS ARRAY(MAP(VARCHAR, VARCHAR))
        ) AS grants_array
    FROM your_table_name
)
SELECT key,
       grants_array,
       grant
FROM grants, UNNEST(grants_array) AS t(grant)
WHERE element_at(grant, 'canonicalId') = 'user-canonical-id';
```

```
# Get the number of grantees on the Object ACL.
SELECT key,
       object_access_control_list,
       json_array_length(json_extract(object_access_control_list,'$.grants')) AS
 grants_count
FROM your_table_name;
```

Untuk informasi lebih lanjut tentang menggunakan Athena, lihatPanduan Pengguna Amazon Athena.

Mengonversi string ID versi kosong di laporan Inventaris Amazon S3 ke string nol

> **Note**
>
> Prosedur berikut hanya berlaku untuk laporan Inventaris Amazon S3 yang menyertakan semua versi, dan hanya jika laporan “semua versi” digunakan sebagai manifes untuk Operasi Batch S3 pada bucket yang mengaktifkan Versi S3. Anda tidak diharuskan mengonversi string untuk laporan S3 Inventory yang menentukan versi saat ini saja.

Anda dapat menggunakan laporan S3 Inventory sebagai manifes untuk Operasi Batch S3. Namun, ketika Versi S3 diaktifkan pada bucket, laporan S3 Inventory yang menyertakan semua versi menandai objek berversi null dengan string kosong di bidang ID versi. Ketika Laporan Inventaris mencakup semua ID versi objek, Operasi Batch mengenali`null`string sebagai ID versi, tetapi tidak string kosong.

Ketika pekerjaan Operasi Batch S3 menggunakan laporan S3 Inventaris “semua versi” sebagai manifes, gagal semua tugas pada objek yang memiliki string kosong di bidang ID versi. Untuk mengonversi string kosong di bidang ID versi laporan S3 Inventory ke`null`string untuk Operasi Batch, gunakan prosedur berikut.

Memperbarui laporan Inventaris Amazon S3 untuk digunakan dengan Operasi Batch

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Arahkan ke laporan S3 Inventory Anda. Laporan inventaris terletak di bucket tujuan yang Anda tentukan saat mengonfigurasi laporan inventaris. Untuk informasi selengkapnya tentang menemukan laporan inventaris, lihatMenemukan daftar inventaris.
   a. Pilih ember tujuan.
   b. Pilih folder. Folder ini dinamai menurut bucket sumber asli.
   c. Pilih folder yang dinamai sesuai konfigurasi inventaris.
   d. Pilih kotak centang di samping folder bernamasarang. Di bagian atas halaman, pilihSalin URI S3untuk menyalin URI S3 untuk folder.
3. Buka konsol Amazon Athena dihttps://console.aws.amazon.com/athena/.

4. Di editor kueri, pilihPengaturan, lalu pilihMengelola. PadaMengelola pengaturanhalaman, untukLokasi hasil kueri, pilih bucket S3 untuk menyimpan hasil kueri Anda.
5. Di editor kueri, buat tabel Athena untuk menyimpan data dalam laporan inventaris menggunakan perintah berikut. Ganti*table_name*dengan nama yang Anda pilih, dan di`LOCATION`klausa, masukkan URI S3 yang Anda salin sebelumnya. Kemudian pilihJalankanuntuk menjalankan query.

```
CREATE EXTERNAL TABLE table_name(bucket string, key string,
 version_id string) PARTITIONED BY (dt string)ROW FORMAT SERDE
 'org.apache.hadoop.hive.serde2.OpenCSVSerde'STORED AS INPUTFORMAT
 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat' OUTPUTFORMAT
 'org.apache.hadoop.hive.ql.io.IgnoreKeyTextOutputFormat' LOCATION 'Copied S3 URI';
```

6. Untuk menghapus editor kueri, pilihJelas. Kemudian muat laporan inventaris ke dalam tabel menggunakan perintah berikut. Ganti*table_name*dengan salah satu yang Anda pilih di langkah sebelumnya. Kemudian pilihJalankanuntuk menjalankan query.

```
MSCK REPAIR TABLE table_name;
```

7. Untuk menghapus editor kueri, pilihJelas. Jalankan yang berikut`SELECT`query untuk mengambil semua entri dalam laporan inventaris asli dan mengganti ID versi kosong dengan`null`senar. Ganti*table_name*dengan yang Anda pilih sebelumnya, dan ganti*YYYY-MM-DD-HH-MM*di dalam`WHERE`klausa dengan tanggal laporan inventaris yang Anda inginkan alat ini berjalan. Kemudian pilihJalankanuntuk menjalankan query.

```
SELECT bucket as Bucket, key as Key, CASE WHEN version_id = '' THEN 'null' ELSE
 version_id END as VersionId FROM table_name WHERE dt = 'YYYY-MM-DD-HH-MM';
```

8. Kembali ke konsol Amazon S3 (https://console.aws.amazon.com/s3/), dan arahkan ke bucket S3 yang Anda pilihLokasi hasil kuerisebelumnya. Di dalam, harus ada serangkaian folder yang diakhiri dengan tanggal.

   Misalnya, Anda harus melihat sesuatu sepertis3://***DOC-CONTOH-EMBER***/***query-result-location***/Tidak disimpan/2021/10/07/. Anda harus melihat`.csv`file yang berisi hasil`SELECT`query bahwa Anda berlari.

   Pilih file CSV dengan tanggal modifikasi terbaru. Unduh file ini ke komputer lokal Anda untuk langkah selanjutnya.

9. File CSV yang dihasilkan berisi baris header. Untuk menggunakan file CSV ini sebagai masukan untuk pekerjaan Operasi Batch S3, Anda harus menghapus baris header, karena Operasi Batch tidak mendukung baris header pada manifes CSV.

   Untuk menghapus baris header, Anda dapat menjalankan salah satu perintah berikut pada file. Ganti*file.csv*dengan nama file CSV Anda.

   Untuk mesin macOS dan Linux, jalankan`tail`perintah di jendela Terminal.

   ```
   tail -n +2 file.csv > tmp.csv && mv tmp.csv file.csv
   ```

   Untuk mesin Windows, jalankan skrip berikut di WindowsPowerShelljendela. Ganti*File-location*dengan path ke file Anda, dan*file.csv*dengan nama file.

   ```
   $ins = New-Object System.IO.StreamReader File-location\file.csv
   $outs = New-Object System.IO.StreamWriter File-location\temp.csv
   try {
       $skip = 0
       while ( !$ins.EndOfStream ) {
           $line = $ins.ReadLine();
           if ( $skip -ne 0 ) {
               $outs.WriteLine($line);
           } else {
               $skip = 1
           }
       }
   } finally {
       $outs.Close();
       $ins.Close();
   }
   Move-Item File-location\temp.csv File-location\file.csv -Force
   ```

10. Setelah menghapus baris header dari file CSV, Anda siap menggunakannya sebagai manifes dalam pekerjaan Operasi Batch S3. Unggah file CSV ke bucket S3 atau lokasi yang Anda pilih, lalu buat tugas Operasi Batch menggunakan file CSV sebagai manifes.

    Untuk informasi selengkapnya tentang membuat pekerjaan Batch Operations, lihat[Membuat pekerjaan Operasi Batch S3](#).

Menggunakan bidang ACL Objek

Laporan Inventaris Amazon S3 berisi daftar objek dalam bucket sumber S3 dan metadata untuk setiap objek. Bidang daftar kontrol akses (ACL) Objek adalah bidang metadata yang tersedia di Inventaris Amazon S3. Secara khusus, ACL Objek berisi daftar kontrol akses (ACL) untuk setiap objek. ACL untuk objek mendefinisikan Akun AWS atau kelompok mana yang diberikan akses ke objek ini dan jenis akses yang diberikan. Untuk informasi selengkapnya, lihat Gambaran umum daftar kontrol akses (ACL) dan Daftar Inventaris Amazon S3.

Bidang ACL Objek dalam laporan Inventaris Amazon S3 ditentukan dalam format JSON. Data JSON terdiri dari bidang berikut:

- `version`–Versi format bidang ACL Objek dalam laporan inventaris. Ini menggunakan format tanggal `yyyy-mm-dd`.
- `status`–Nilai yang memungkinkan adalah `AVAILABLE` atau `UNAVAILABLE` untuk menunjukkan apakah ACL Objek tersedia untuk suatu objek. Ketika status untuk ACL Objek adalah `UNAVAILABLE`, nilai bidang Pemilik Objek dalam laporan inventaris juga `UNAVAILABLE`.
- `grants`–Pasangan izin-penerima yang mencantumkan status izin dari setiap penerima izin yang diberikan oleh ACL Objek. Nilai yang tersedia untuk penerima izin adalah `CanonicalUser` dan `Group`. Untuk informasi selengkapnya tentang penerima izin, lihat Penerima izin dalam daftar kontrol akses.

  Untuk penerima izin dengan tipe `Group`, pasangan izin-penerima mencakup atribut berikut:

  - `uri`–Grup Amazon S3 yang telah ditentukan sebelumnya.
  - `permission`–Izin ACL yang diberikan pada objek. Untuk informasi selengkapnya, lihat izin ACL pada objek.
  - `type`–Jenis `Group`, yang menunjukkan bahwa penerima izin adalah kelompok.

  Untuk penerima izin dengan jenis `CanonicalUser`, pasangan izin-penerima mencakup atribut berikut:

  - `canonicalId`–Bentuk ID Akun AWS yang disamarkan. ID pengguna kanonik untuk akun khusus untuk akun Akun AWS itu. Anda dapat mengambil ID pengguna kanonik. Untuk informasi selengkapnya, lihat Temukan ID pengguna kanonik untuk Anda Akun AWS di Panduan Referensi Manajemen AWS Akun.

> **Note**
>
> Jika penerima hibah dalam ACL adalah alamat email dari sebuah Akun AWS, S3 Inventory menggunakan dari itu Akun AWS dan `CanonicalUser` jenisnya untuk menentukan penerima hibah ini. `canonicalId` Untuk informasi selengkapnya, lihat Penerima izin dalam daftar kontrol akses.

- `permission`–Izin ACL yang diberikan pada objek. Untuk informasi selengkapnya, lihat izin ACL pada objek.
- `type`— Jenis`CanonicalUser`, yang menunjukkan bahwa penerima hibah adalah. Akun AWS

Contoh berikut menunjukkan nilai yang memungkinkan untuk bidang ACL Objek dalam format JSON:

```
{
    "version": "2022-11-10",
    "status": "AVAILABLE",
    "grants": [{
        "uri": "http://acs.amazonaws.com/groups/global/AllUsers",
        "permission": "READ",
        "type": "Group"
    }, {
        "canonicalId": "example-canonical-id",
        "permission": "FULL_CONTROL",
        "type": "CanonicalUser"
    }]
}
```

> **Note**
>
> Bidang ACL Objek ditentukan dalam format JSON. Laporan inventaris akan menampilkan nilai untuk bidang Object ACL sebagai string yang dikodekan base64.
> Misalnya, katakanlah Anda memiliki ACL Objek berikut dalam format JSON:
>
> ```
> {
>  "version": "2022-11-10",
>  "status": "AVAILABLE",
>  "grants": [{
>  "canonicalId": "example-canonical-user-ID",
>  "type": "CanonicalUser",
> ```

```
            "permission": "READ"
        }]
}
```

Bidang ACL Objek dikodekan dan ditampilkan sebagai string yang dikodekan dengan base64 berikut ini:


```
eyJ2ZXJzaW9uIjoiMjAyMi0xMS0xMCIsInN0YXR1cyI6IkFWQUlMQUJMRSIsImdyYW50cyI6W3siY2Fub25pY2FsSW
```


Untuk mendapatkan nilai yang terdekripsi dalam JSON untuk bidang ACL Objek, Anda dapat melakukan kueri pada bidang ini di Amazon Athena. Untuk contoh kueri, lihat Menanyakan Inventaris Amazon S3 dengan Amazon Athena.

Mereplikasi ikhtisar objek

Anda dapat menggunakan replikasi untuk mengaktifkan penyalinan objek secara otomatis dan asinkron di seluruh bucket Amazon S3. Bucket yang dikonfigurasi untuk replikasi objek dapat dimiliki oleh Akun AWS yang sama atau oleh akun yang berbeda. Anda dapat mereplikasi objek ke satu bucket tujuan atau ke beberapa bucket tujuan. Bucket tujuan bisa berbeda Wilayah AWS atau dalam Wilayah yang sama dengan ember sumber.

Ada dua jenis replikasi: replikasi langsung dan replikasi sesuai permintaan.

- Replikasi langsung — Untuk secara otomatis mereplikasi objek baru dan yang diperbarui saat ditulis ke keranjang sumber, gunakan replikasi langsung. Replikasi langsung tidak mereplikasi objek apa pun yang ada di bucket sebelum Anda mengatur replikasi. Untuk mereplikasi objek yang ada sebelum Anda mengatur replikasi, gunakan replikasi sesuai permintaan.
- Replikasi sesuai permintaan — Untuk mereplikasi objek yang ada dari bucket sumber ke satu atau beberapa bucket tujuan sesuai permintaan, gunakan Replikasi Batch S3. Untuk informasi selengkapnya tentang mereplikasi objek yang ada, lihat Kapan menggunakan Replikasi Batch S3.

Ada dua bentuk replikasi hidup: Replikasi Lintas Wilayah (CRR) dan Replikasi Wilayah Tunggal (SRR).

- Replikasi Lintas Wilayah (CRR) - Anda dapat menggunakan CRR untuk mereplikasi objek di seluruh bucket Amazon S3 secara berbeda. Wilayah AWS Untuk informasi lebih lanjut tentang CRR, lihatthe section called “Kapan menggunakan Replikasi Lintas-Wilayah”.

- Replikasi Wilayah Tunggal (SRR) - Anda dapat menggunakan SRR untuk menyalin objek di bucket Amazon S3 secara bersamaan. Wilayah AWS Untuk informasi lebih lanjut tentang SRR, lihatthe section called “Kapan menggunakan Replikasi Lintas-Wilayah”.

Topik

- Mengapa menggunakan replikasi?
- Kapan menggunakan Replikasi Lintas-Wilayah
- Kapan menggunakan Replikasi Lintas-Wilayah
- Kapan menggunakan replikasi dua arah (replikasi bidireksional)
- Kapan menggunakan Replikasi Batch S3
- Persyaratan beban kerja dan replikasi langsung
- Apa yang direplikasi Amazon S3?
- Persyaratan dan pertimbangan untuk replikasi
- Menyiapkan replikasi langsung
- Mengelola atau menjeda replikasi langsung
- Memantau progres dengan metrik replikasi dan Notifikasi Peristiwa S3
- Mereplikasi objek yang ada dengan Replikasi Batch S3

Mengapa menggunakan replikasi?

Replikasi dapat membantu Anda melakukan hal berikut:

- Replikasi objek sambil mempertahankan metadata–Anda dapat menggunakan replikasi untuk membuat salinan objek yang mempertahankan semua metadata, seperti waktu pembuatan objek asli dan ID versi. Kemampuan ini penting jika Anda harus memastikan bahwa replika Anda identik dengan objek sumber.
- Replikasi objek ke dalam kelas penyimpanan yang berbeda–Anda dapat menggunakan replikasi untuk secara langsung memasukkan objek ke dalam S3 Glacier Flexible Retrieval, S3 Glacier Deep Archive, atau kelas penyimpanan lainnya di bucket tujuan. Anda juga dapat mereplikasi data Anda ke kelas penyimpanan yang sama dan menggunakan konfigurasi siklus hidup pada bucket tujuan untuk memindahkan objek Anda ke kelas penyimpanan yang lebih dingin seiring bertambahnya usia.

- Pertahankan salinan objek di bawah kepemilikan yang berbeda — Terlepas dari siapa yang memiliki objek sumber, Anda dapat memberi tahu Amazon S3 untuk mengubah kepemilikan replika ke yang memiliki Akun AWS bucket tujuan. Ini disebut sebagai opsi penggantian pemilik. Anda dapat menggunakan opsi ini untuk membatasi akses ke replika objek.
- Simpan objek yang disimpan di beberapa tempat Wilayah AWS— Untuk memastikan perbedaan geografis di mana data Anda disimpan, Anda dapat mengatur beberapa bucket tujuan di berbagai tempat. Wilayah AWS Fitur ini dapat membantu Anda memenuhi persyaratan kepatuhan tertentu.
- Replikasi objek dalam waktu 15 menit — Untuk mereplikasi data Anda di Wilayah yang sama Wilayah AWS atau di berbagai Wilayah dalam jangka waktu yang dapat diprediksi, Anda dapat menggunakan Kontrol Waktu Replikasi S3 (S3 RTC). S3 RTC mereplikasi 99,99 persen objek baru yang disimpan di Amazon S3 dalam 15 menit (didukung oleh perjanjian tingkat layanan). Untuk informasi selengkapnya, lihat the section called “Menggunakan Kontrol Waktu Replikasi S3”.

> **Note**
>
> S3 RTC tidak berlaku untuk Replikasi Batch. Replikasi Batch adalah pekerjaan replikasi sesuai permintaan, dan dapat dilacak dengan Operasi Batch S3. Untuk informasi selengkapnya, lihat Melacak status tugas dan laporan penyelesaian.

- Menyinkronkan bucket, mereplikasi objek yang ada, dan mereplikasi objek yang sebelumnya gagal atau direplikasi–Untuk menyinkronkan bucket dan mereplikasi objek yang ada, gunakan Replikasi Batch sebagai tindakan replikasi sesuai permintaan. Untuk informasi selengkapnya tentang kapan menggunakan Replikasi Batch, lihat Kapan menggunakan Replikasi Batch S3.
- Replikasi objek dan failover ke bucket di Wilayah AWS lain–Untuk menjaga agar semua metadata dan objek tetap sinkron di seluruh bucket selama replikasi data, gunakan aturan replikasi dua arah (juga dikenal sebagai replikasi bidireksional) sebelum mengonfigurasi kontrol failover Titik Akses Multi-Wilayah Amazon S3. Aturan replikasi dua arah membantu memastikan bahwa ketika data ditulis ke bucket S3 yang lalu lintas failover, data tersebut kemudian direplikasi kembali ke bucket sumber.

Kapan menggunakan Replikasi Lintas-Wilayah

Replikasi Lintas Wilayah (CRR) S3 digunakan untuk menyalin objek di seluruh bucket Amazon S3 di Wilayah AWS yang berbeda. CRR dapat membantu Anda melakukan ini:

- Memenuhi persyaratan kepatuhan–Meskipun Amazon S3 menyimpan data Anda di beberapa Zona Ketersediaan yang secara geografis berjauhan secara default, persyaratan kepatuhan mungkin mengatur bahwa Anda menyimpan data dengan jarak yang lebih jauh. Untuk memenuhi persyaratan ini, gunakan Replikasi Lintas Wilayah untuk mereplikasi data antara Wilayah AWS yang jauh.
- Minimalkan latensi — Jika pelanggan Anda berada di dua lokasi geografis, Anda dapat meminimalkan latensi dalam mengakses objek dengan mempertahankan salinan objek Wilayah AWS yang secara geografis lebih dekat dengan pengguna Anda.
- Tingkatkan efisiensi operasional — Jika Anda memiliki cluster komputasi dalam dua kelompok berbeda Wilayah AWS yang menganalisis kumpulan objek yang sama, Anda dapat memilih untuk mempertahankan salinan objek di Wilayah tersebut.

Kapan menggunakan Replikasi Lintas-Wilayah

Replikasi Wilayah yang Sama (SRR) digunakan untuk menyalin objek di seluruh bucket Amazon S3 dalam Wilayah AWS yang sama. CRR dapat membantu Anda melakukan ini:

- Menggabungkan log ke dalam satu bucket–Jika Anda menyimpan log dalam beberapa bucket atau beberapa akun, Anda dapat dengan mudah mereplikasi log ke dalam satu bucket di Wilayah. Melakukan hal itu memungkinkan pemrosesan log yang lebih sederhana di satu lokasi.
- Mengonfigurasi replikasi langsung antara akun produksi dan pengujian–Jika Anda atau pelanggan Anda memiliki akun produksi dan pengujian yang menggunakan data yang sama, Anda dapat mereplikasi objek di antara beberapa akun tersebut, sambil mempertahankan metadata objek.
- Mematuhi undang-undang kedaulatan data — Anda mungkin diminta untuk menyimpan beberapa salinan data Anda secara terpisah Akun AWS dalam Wilayah tertentu. Replikasi di Wilayah yang sama dapat membantu Anda mereplikasi data penting secara otomatis ketika peraturan kepatuhan tidak mengizinkan data meninggalkan negara Anda.

Kapan menggunakan replikasi dua arah (replikasi bidireksional)

- Membangun kumpulan data bersama di beberapa Wilayah AWS — Dengan sinkronisasi modifikasi replika, Anda dapat dengan mudah mereplikasi perubahan metadata, seperti daftar kontrol akses objek (ACL), tag objek, atau kunci objek, pada objek replikasi. Replikasi dua arah ini penting jika Anda ingin menjaga agar semua objek dan perubahan metadata objek tetap sinkron. Anda dapat mengaktifkan sinkronisasi modifikasi replika pada aturan replikasi baru atau yang sudah ada saat

melakukan replikasi dua arah antara dua bucket atau lebih dalam Wilayah AWS yang sama atau berbeda.

- Menyinkronkan data di seluruh Wilayah selama failover — Anda dapat menyinkronkan data dalam bucket di antaranya Wilayah AWS dengan mengonfigurasi aturan replikasi dua arah dengan S3 Cross-Region Replication (CRR) langsung dari Titik Akses Multi-Wilayah. Untuk membuat keputusan berdasarkan informasi tentang kapan memulai failover, Anda juga dapat mengaktifkan metrik replikasi S3 sehingga Anda dapat memantau replikasi di Amazon CloudWatch, di Kontrol Waktu Replikasi S3 (S3 RTC), atau dari Titik Akses Multi-Wilayah.
- Menjadikan aplikasi Anda sangat tersedia–Bahkan jika terjadi gangguan lalu lintas Regional, Anda dapat menggunakan aturan replikasi dua arah untuk menjaga agar semua metadata dan objek tetap sinkron di seluruh bucket selama replikasi data.

Kapan menggunakan Replikasi Batch S3

Replikasi Batch mereplikasi objek yang ada ke bucket yang berbeda sebagai opsi sesuai permintaan. Tidak seperti replikasi langsung, pekerjaan ini dapat dijalankan sesuai kebutuhan. Replikasi Batch dapat membantu Anda melakukan hal berikut:

- Mereplikasi objek yang ada–Anda dapat menggunakan Replikasi Batch untuk mereplikasi objek yang ditambahkan ke bucket sebelum Replikasi Wilayah yang Sama atau Replikasi Lintas Wilayah dikonfigurasi.
- Mereplikasi objek yang sebelumnya gagal direplikasi–Anda dapat memfilter pekerjaan Replikasi Batch untuk mencoba mereplikasi objek dengan status replikasi GAGAL.
- Replikasi objek yang sudah direplikasi–Anda mungkin diminta untuk menyimpan beberapa salinan data Anda di Akun AWS atau Wilayah AWS terpisah. Replikasi Batch dapat mereplikasi objek yang ada ke tujuan yang baru ditambahkan.
- Replikasi replika objek yang dibuat dari aturan replikasi–Konfigurasi replikasi membuat replika objek dalam bucket tujuan. Replika objek hanya dapat direplikasi dengan Replikasi Batch.

Persyaratan beban kerja dan replikasi langsung

Bergantung pada persyaratan beban kerja Anda, beberapa jenis replikasi akan lebih cocok untuk kasus penggunaan Anda daripada yang lain. Gunakan tabel berikut untuk menentukan jenis replikasi yang akan digunakan untuk situasi Anda, dan apakah akan menggunakan Kontrol Waktu Replikasi S3 (S3 RTC) untuk beban kerja Anda. S3 RTC mereplikasi 99,99 persen objek baru yang disimpan

di Amazon S3 dalam waktu 15 menit (didukung oleh perjanjian tingkat layanan, atau SLA). Untuk informasi selengkapnya, lihat the section called “Menggunakan Kontrol Waktu Replikasi S3”.

Persyaratan beban kerja untuk perbandingan replikasi

Persyaratan beban kerja	S3 RTC (15 menit SLA)	Replikasi Lintas Wilayah (CRR)	Replikasi Wilayah Tunggal (SRR)
Replikasi objek antara yang berbeda Akun AWS	Ya	Ya	Ya
Replikasi objek dalam waktu yang sama Wilayah AWS dalam 24-48 jam (tidak didukung SLA)	Tidak	Tidak	Ya
Replikasi objek antara yang berbeda Wilayah AWS dalam 24-48 jam (tidak didukung SLA)	Tidak	Ya	Tidak
Waktu replikasi yang dapat diprediksi: Didukung oleh SLA untuk mereplikasi 99,9 persen objek dalam 15 menit	Ya	Tidak	Tidak

Apa yang direplikasi Amazon S3?

Amazon S3 hanya mereplikasi item tertentu dalam bucket yang dikonfigurasi untuk replikasi.

Topik

- Apa yang direplikasi dengan konfigurasi replikasi?
- Apa yang tidak direplikasi dengan konfigurasi replikasi?

- Bagaimana enkripsi bucket default memengaruhi replikasi

Apa yang direplikasi dengan konfigurasi replikasi?

Secara default, Amazon S3 mereplikasi berikut ini:

- Objek yang dibuat setelah Anda menambahkan konfigurasi replikasi.
- Objek tidak terenkripsi.
- Objek yang dienkripsi menggunakan kunci yang disediakan pelanggan (SSE-C), objek yang dienkripsi saat istirahat di bawah kunci terkelola Amazon S3 (SSE-S3) atau kunci KMS yang disimpan di (SSE-KMS). AWS Key Management Service Untuk informasi selengkapnya, lihat the section called “Mereplikasi objek terenkripsi”.
- Metadata objek dari objek sumber ke replika. Untuk informasi tentang mereplikasi metadata dari replika ke objek sumber, lihat Mereplikasi perubahan metadata dengan sinkronisasi modifikasi replika Amazon S3.
- Hanya objek dalam bucket sumber yang diizinkan oleh pemilik bucket untuk membaca objek dan daftar kontrol akses (ACL).

  Untuk informasi selengkapnya tentang kepemilikan sumber daya, lihat bucket Amazon S3 dan kepemilikan objek.
- Object ACL diperbarui, kecuali jika Anda mengarahkan Amazon S3 untuk mengubah kepemilikan replika ketika bucket sumber dan tujuan tidak dimiliki oleh akun yang sama.

  Untuk informasi selengkapnya, lihat Mengubah pemilik replika.

  Perlu beberapa waktu sampai Amazon S3 bisa menyeleraskan kedua ACL. Perubahan kepemilikan ini hanya berlaku untuk objek yang dibuat setelah Anda menambahkan konfigurasi replikasi ke bucket.
- Tag objek, jika ada.
- Informasi retensi Kunci Objek S3, jika ada.

  Saat Amazon S3 mereplikasi objek yang memiliki informasi retensi yang berlaku, itu akan menerapkan kontrol retensi yang sama ke replika Anda, mengabaikan periode retensi default yang dikonfigurasi pada bucket tujuan Anda. Jika Anda tidak memiliki kontrol retensi yang diterapkan pada objek di bucket sumber, dan Anda mereplikasi ke dalam bucket tujuan yang sudah ditetapkan dengan periode retensi default, periode retensi default bucket tujuan diterapkan ke replika objek Anda. Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek S3.

Bagaimana cara menghapus operasi yang memengaruhi replikasi

Jika Anda menghapus objek dari bucket sumber, tindakan berikut terjadi secara default:

- Jika Anda membuat permintaan DELETE tanpa menentukan ID versi objek, Amazon S3 akan menambahkan penanda hapus. Amazon S3 menangani penanda hapus sebagai berikut:
  - Jika Anda menggunakan versi terbaru konfigurasi replikasi (yaitu, Anda menentukan elemen `Filter` pada aturan konfigurasi replikasi), Amazon S3 tidak mereplikasi penanda hapus secara default. Namun, Anda dapat menambahkan replikasi penanda hapus ke non-tag-based aturan. Untuk informasi selengkapnya, lihat Mereplikasi penanda hapus di antara bucket.
  - Jika Anda tidak menentukan elemen `Filter`, Amazon S3 berasumsi bahwa konfigurasi replikasi adalah versi V1 dan mereplikasi penanda hapus yang dihasilkan dari tindakan pengguna. Namun, jika Amazon S3 menghapus objek karena tindakan siklus hidup, penanda hapus tidak direplikasi ke bucket tujuan.
- Jika Anda menentukan ID versi objek yang akan dihapus dalam permintaan `DELETE`, Amazon S3 akan menghapus versi objek tersebut di bucket sumber. Tetapi ini tidak mereplikasi penghapusan di bucket tujuan. Dengan kata lain, itu tidak menghapus versi objek yang sama dari bucket tujuan. Ini melindungi data dari penghapusan berbahaya.

Apa yang tidak direplikasi dengan konfigurasi replikasi?

Secara default, Amazon S3 mereplikasi berikut ini:

- Objek dalam bucket sumber yang merupakan replika yang dibuat oleh aturan replikasi lain. Misalnya, Anda mengonfigurasi replikasi dengan bucket A sebagai sumber dan bucket B sebagai tujuan. Sekarang bayangkan bahwa Anda menambahkan konfigurasi replikasi lain dengan bucket B sebagai sumber dan bucket C sebagai tujuan. Dalam hal ini, objek dalam bucket B yang merupakan replika objek dalam bucket A tidak direplikasi ke bucket C.

  Untuk mereplikasi objek yang merupakan replika, gunakan Replikasi Batch. Pelajari lebih lanjut mengonfigurasi Replikasi Batch di Mereplikasi objek yang ada.
- Objek dalam bucket sumber yang telah direplikasi ke tujuan yang berbeda. Misalnya, jika Anda mengubah bucket tujuan dalam konfigurasi replikasi yang ada, Amazon S3 tidak akan mereplikasi objek itu lagi.

  Untuk mereplikasi objek yang direplikasi sebelumnya, gunakan Replikasi Batch. Pelajari lebih lanjut mengonfigurasi Replikasi Batch di Mereplikasi objek yang ada.

- Replikasi Batch tidak mendukung replikasi ulang objek yang telah dihapus dengan ID versi objek dari bucket tujuan. Untuk mereplikasi ulang objek ini, Anda dapat menyalin objek sumber di tempat dengan tugas Penyalinan Batch. Menyalin objek tersebut di tempat akan membuat versi baru objek di bucket sumber dan memulai replikasi secara otomatis ke tujuan. Untuk informasi selengkapnya tentang cara menggunakan Penyalinan Batch, lihat, Contoh yang menggunakan Operasi Batch untuk menyalin objek.
- Secara default, saat mereplikasi dari yang berbeda Akun AWS, penanda hapus yang ditambahkan ke bucket sumber tidak akan direplikasi.

  Untuk informasi tentang cara mereplikasi penanda hapus, lihat Mereplikasi penanda hapus di antara bucket.
- Objek yang disimpan di S3 Glacier Flexible Retrieval, S3 Glacier Deep Archive, S3 Intelligent-Tiering Archive Access, atau kelas penyimpanan S3 Intelligent-Tiering Deep Archive Access atau tingkatan. Anda tidak dapat mereplikasi objek ini sampai Anda mengembalikannya dan menyalinnya ke kelas penyimpanan yang berbeda.

  Untuk mempelajari lebih lanjut tentang S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive, lihat. Kelas penyimpanan untuk objek yang jarang diakses

  Untuk mempelajari lebih lanjut tentang S3 Intelligent-Tiering, lihat. Amazon S3 Intelligent-Tiering
- Objek di bucket sumber yang pemilik bucket-nya tidak memiliki izin yang cukup untuk replikasi.

  Untuk informasi tentang bagaimana pemilik objek dapat memberikan izin kepada pemilik bucket, lihat Berikan izin lintas akun untuk unggah objek sekaligus memastikan bahwa pemilik bucket memiliki kendali penuh.
- Pembaruan untuk subsumber daya tingkat bucket.

  Misalnya, jika Anda mengubah konfigurasi siklus aktif atau menambahkan konfigurasi pemberitahuan ke bucket sumber Anda, perubahan ini tidak diterapkan ke bucket tujuan. Fitur ini memungkinkan untuk memiliki konfigurasi berbeda pada bucket sumber dan tujuan.
- Tindakan yang dilakukan berdasarkan konfigurasi siklus aktif.

  Misalnya, jika konfigurasi siklus hidup diaktifkan hanya pada bucket sumber Anda, Amazon S3 membuat penanda hapus untuk objek yang kedaluwarsa tetapi tidak mereplikasi penanda tersebut. Jika Anda ingin konfigurasi siklus aktif yang sama diterapkan pada bucket sumber dan tujuan, aktifkan konfigurasi siklus aktif yang sama pada keduanya. Untuk informasi lebih lanjut tentang konfigurasi siklus aktif, lihat Mengelola siklus hidup penyimpanan Anda.

- Saat Anda menggunakan aturan replikasi berbasis tag dengan replikasi langsung, objek baru harus ditandai dengan tag aturan replikasi yang cocok dalam operasi. `PutObject` Jika tidak, objek tidak akan direplikasi. Jika objek diberi tag setelah `PutObject` operasi, objek tersebut juga tidak akan direplikasi.

  Untuk mereplikasi objek yang telah ditandai setelah `PutObject` operasi, Anda harus menggunakan Replikasi Batch S3. Untuk informasi selengkapnya tentang Replikasi Batch, lihat Mereplikasi objek yang ada.

Bagaimana enkripsi bucket default memengaruhi replikasi

Saat Anda mengaktifkan enkripsi default untuk bucket tujuan replikasi, perilaku enkripsi berikut berlaku:

- Jika objek dalam bucket sumber tidak dienkripsi, objek replika dalam bucket tujuan akan dienkripsi menggunakan pengaturan enkripsi default dari bucket tujuan. Akibatnya, tag entitas (ETag) dari objek sumber berbeda dari ETag objek replika. Jika Anda memiliki aplikasi yang menggunakan ETag, Anda harus memperbarui aplikasi tersebut untuk memperhitungkan perbedaan ini.
- Jika objek dalam bucket sumber dienkripsi dengan menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3), enkripsi sisi server dengan kunci () (SSE-KMS), atau enkripsi sisi server dua lapis dengan AWS Key Management Service kunci (DSSE-KMS), objek replika di bucket tujuan menggunakan jenis enkripsi yang sama dengan AWS KMS objek sumber.AWS KMS Pengaturan enkripsi default bucket tujuan tidak digunakan.

Persyaratan dan pertimbangan untuk replikasi

Replikasi Amazon S3 membutuhkan yang berikut:

- Pemilik bucket sumber harus Wilayah AWS mengaktifkan sumber dan tujuan untuk akun mereka. Pemilik bucket tujuan harus mengaktifkan Wilayah tujuan untuk akun mereka.

  Untuk informasi selengkapnya tentang mengaktifkan atau menonaktifkan Wilayah AWS, lihat Mengelola Wilayah AWS di. Referensi Umum AWS
- Bucket sumber dan tujuan harus memiliki Penentuan Versi aktif. Untuk informasi selengkapnya tentang penentuan versi, lihat Menggunakan Penentuan Versi dalam bucket S3.

- Amazon S3 harus memiliki izin untuk mereplikasi objek dari bucket sumber ke bucket tujuan atau bucket atas nama Anda. Untuk informasi selengkapnya tentang izin ini, lihat Menyiapkan izin untuk replikasi langsung.
- Apabila pemilik bucket sumber tidak memiliki objek dalam bucket, pemilik objek harus memberi pemilik bucket izin `READ` dan `READ_ACP` dengan daftar kontrol akses (ACL) objek. Untuk informasi selengkapnya, lihat Gambaran umum daftar kontrol akses (ACL).
- Jika bucket sumber mengaktifkan Kunci Objek S3, bucket tujuan juga harus mengaktifkan Kunci Objek S3.

  Untuk mengaktifkan replikasi pada bucket yang mengaktifkan Object Lock, Anda harus menggunakan AWS Command Line Interface, REST API, atau AWS SDK. Untuk informasi umum selengkapnya, lihat Menggunakan Kunci Objek S3.

> **Note**
>
> Anda harus memberikan dua izin baru pada bucket S3 sumber dalam peran AWS Identity and Access Management (IAM) yang Anda gunakan untuk menyiapkan replikasi. Dua izin baru adalah `s3:GetObjectRetention` dan `s3:GetObjectLegalHold`. Jika peran memiliki izin `s3:Get*`, itu memenuhi persyaratan. Untuk informasi selengkapnya, lihat Menyiapkan izin untuk replikasi langsung.

Untuk informasi selengkapnya, lihat Menyiapkan replikasi langsung.

Jika Anda mengatur konfigurasi replikasi dalam skenario lintas akun, yang bucket sumber dan tujuannya dimiliki oleh Akun AWS yang berbeda, persyaratan tambahan berikut ini berlaku:

- Pemilik bucket tujuan harus memberikan izin kepada pemilik bucket sumber untuk mereplikasi objek dengan kebijakan bucket. Untuk informasi selengkapnya, lihat Memberikan izin saat bucket sumber dan tujuan dimiliki oleh yang berbeda Akun AWS.
- Bucket tujuan tidak dapat dikonfigurasi sebagai bucket Pembayaran oleh Pemohon. Untuk informasi selengkapnya, lihat Menggunakan bucket Pembayaran Pemohon untuk transfer dan penggunaan penyimpanan.

Pertimbangan untuk replikasi

Sebelum Anda membuat konfigurasi replikasi, perhatikan pertimbangan berikut.

Topik

- Konfigurasi siklus hidup dan replika objek
- Konfigurasi Penentuan Versi dan konfigurasi replikasi
- Menggunakan Replikasi S3 dengan S3 Intelligent-Tiering
- Konfigurasi pencatatan dan konfigurasi replikasi
- CRR dan wilayah tujuan
- Replikasi Batch S3
- Kontrol Waktu Replikasi S3

Konfigurasi siklus hidup dan replika objek

Waktu yang diperlukan Amazon S3 untuk mereplikasi objek tergantung pada ukuran objek. Untuk objek besar, dibutuhkan waktu beberapa jam. Meskipun mungkin perlu beberapa saat sebelum replika tersedia di tujuan, dibutuhkan waktu yang sama untuk membuat replika seperti yang diperlukan untuk membuat objek yang sesuai di bucket sumber. Jika konfigurasi siklus hidup diaktifkan pada bucket tujuan, aturan siklus hidup menghormati waktu pembuatan asli objek, bukan saat replika tersedia di bucket tujuan.

Konfigurasi replikasi memerlukan bucket dengan dukungan Penentuan Versi. Saat Anda mengaktifkan Penentuan Versi di bucket, ingatlah hal berikut:

- Jika Anda memiliki konfigurasi siklus hidup Kedaluwarsa objek, setelah Anda mengaktifkan Penentuan Versi, tambahkan kebijakan `NonCurrentVersionExpiration` untuk mempertahankan perilaku penghapusan permanen yang sama seperti sebelum Anda mengaktifkan Penentuan Versi.
- Jika Anda memiliki konfigurasi siklus hidup Transisi, setelah Anda mengaktifkan Penentuan Versi, pertimbangkan untuk menambahkan kebijakan `NonCurrentVersionTransition`.

Konfigurasi Penentuan Versi dan konfigurasi replikasi

Bucket sumber dan tujuan harus memiliki dukungan Penentuan Versi ketika Anda mengonfigurasi replikasi pada bucket. Setelah Anda mengaktifkan Penentuan Versi pada bucket sumber dan tujuan dan mengonfigurasi replikasi pada bucket sumber, Anda akan menemui masalah berikut ini:

- Jika Anda mencoba menonaktifkan Penentuan Versi pada bucket sumber, Amazon S3 menampilkan pesan kesalahan. Anda harus menghapus konfigurasi replikasi sebelum Anda dapat menonaktifkan Penentuan Versi pada bucket sumber.
- Jika Anda menonaktifkan Penentuan Versi pada bucket tujuan, replikasi akan gagal. Objek sumber memiliki status replikasi `FAILED`.

Menggunakan Replikasi S3 dengan S3 Intelligent-Tiering

S3 Intelligent-Tiering adalah kelas penyimpanan yang dirancang untuk mengoptimalkan biaya penyimpanan dengan memindahkan data ke tingkat akses yang paling hemat biaya secara otomatis. Untuk biaya pemantauan dan otomatisasi objek bulanan yang murah, S3 Intelligent-Tiering memantau pola akses dan secara otomatis memindahkan objek yang belum diakses ke jenjang akses berbiaya rendah.

Mereplikasi objek yang disimpan dalam S3 Intelligent-Tiering dengan Replikasi Batch S3 atau memanggil CopyObject atau UploadPartCopy merupakan akses. Dalam kasus ini, objek sumber dari operasi salinan atau replikasi naik tingkat.

Untuk informasi selengkapnya tentang S3 Intelligent-Tiering lihat, Amazon S3 Intelligent-Tiering.

Konfigurasi pencatatan dan konfigurasi replikasi

Jika Amazon S3 mengirimkan log ke bucket dengan dukungan replikasi, Amazon S3 mereplikasi objek log.

Jika log akses server (Pencatatan permintaan dengan pencatatan akses server) atau log AWS CloudTrail (Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail) diaktifkan di bucket sumber atau tujuan Anda, Amazon S3 menyertakan permintaan terkait replikasi dalam log. Misalnya, Amazon S3 mencatat setiap objek yang direplikasi.

CRR dan wilayah tujuan

Amazon S3 Cross-Region Replication (CRR) digunakan untuk menyalin objek di bucket S3 secara berbeda. Wilayah AWS Anda dapat memilih Wilayah untuk bucket tujuan Anda berdasarkan kebutuhan bisnis atau pertimbangan biaya Anda. Misalnya, biaya transfer data antar Wilayah berbeda tergantung Wilayah yang Anda pilih.

Misalkan, Anda memilih AS Timur (Virginia Utara) (us-east-1) sebagai Wilayah untuk bucket sumber Anda. Jika Anda memilih AS Barat (Oregon) (us-west-2) sebagai Wilayah untuk bucket tujuan Anda,

Anda membayar lebih banyak daripada jika Anda memilih Wilayah AS Timur (Ohio) (us-east-2). Untuk informasi harga, lihat "Harga Transfer Data" di Harga Amazon S3.

Tidak ada biaya transfer data yang terkait dengan Replikasi Wilayah yang Sama (SRR).

Replikasi Batch S3

Untuk informasi tentang pertimbangan untuk Replikasi Batch, lihat. Pertimbangan Replikasi Batch S3

Kontrol Waktu Replikasi S3

Untuk informasi tentang praktik terbaik dan pertimbangan untuk Kontrol Waktu Replikasi S3 (S3 RTC), lihat. Praktik terbaik dan pedoman untuk S3 RTC

Menyiapkan replikasi langsung

> **Note**
>
> Objek yang ada sebelum Anda menyiapkan replikasi tidak direplikasi secara otomatis. Dengan kata lain, Amazon S3 tidak mereplikasi objek secara retroaktif. Untuk mereplikasi objek yang dibuat sebelum konfigurasi replikasi Anda, gunakan Replikasi Batch S3. Pelajari lebih lanjut mengonfigurasi Replikasi Batch di Mereplikasi objek yang ada.

Untuk mengaktifkan replikasi langsung—Same-region Replication (SRR) atau Cross-Region Replication (CRR) —tambahkan konfigurasi replikasi ke bucket sumber Anda. Konfigurasi ini memberitahu Amazon S3 untuk mereplikasi objek seperti yang ditentukan. Dalam konfigurasi replikasi, Anda harus memberikan berikut ini:

- Bucket tujuan–Bucket tempat Anda ingin Amazon S3 mereplikasi objek.
- Objek yang ingin Anda replikasi–Anda dapat mereplikasi semua objek dalam bucket sumber atau subset. Anda mengidentifikasi subset dengan menyediakan awalan nama kunci, satu atau lebih tag objek, atau keduanya dalam konfigurasi.

  Misalnya, jika Anda mengonfigurasi aturan replikasi agar hanya mereplikasi objek dengan prefiks nama kunci `Tax/`, Amazon S3 akan mereplikasi objek dengan kunci seperti `Tax/doc1` atau `Tax/doc2`. Namun, aturan ini tidak mereplikasi objek dengan kunci `Legal/doc3`. Jika Anda menentukan prefiks dan satu atau beberapa tag, Amazon S3 hanya mereplikasi objek yang memiliki kunci prefiks dan tag tertentu.

- Peran AWS Identity and Access Management (IAM) - Amazon S3 mengasumsikan peran IAM ini untuk mereplikasi objek atas nama Anda.

Selain persyaratan minimum ini, Anda dapat memilih opsi berikut:

- Kelas penyimpanan replika–Secara default, Amazon S3 menyimpan replika objek menggunakan kelas penyimpanan yang sama dengan objek sumber. Anda dapat menentukan kelas penyimpanan yang berbeda untuk replika.
- Kepemilikan replika–Amazon S3 berasumsi bahwa replika objek terus dimiliki oleh pemilik objek sumber. Jadi, ketika mereplikasi objek, ini juga mereplikasi daftar kontrol akses (ACL) objek yang terkait atau pengaturan Kepemilikan Objek S3. Jika bucket sumber dan tujuan dimiliki oleh Akun AWS yang berbeda, Anda dapat mengonfigurasi replikasi untuk mengubah pemilik replika ke Akun AWS yang memiliki bucket tujuan.

Anda dapat mengonfigurasi replikasi dengan menggunakan REST API, AWS SDK, AWS Command Line Interface (AWS CLI), atau konsol Amazon S3.

Amazon S3 juga menyediakan operasi API untuk mendukung pengaturan aturan replikasi. Untuk informasi selengkapnya, lihat topik berikut di Referensi API Amazon Simple Storage Service:

- PutBucketReplication
- GetBucketReplication
- DeleteBucketReplication

Topik

- Konfigurasi Replikasi
- Menyiapkan izin untuk replikasi langsung
- Contoh untuk mengonfigurasi replikasi langsung

Konfigurasi Replikasi

Amazon S3 menyimpan konfigurasi replikasi sebagai XML. Dalam file XHTML konfigurasi replikasi, Anda menentukan peran AWS Identity and Access Management (IAM) dan satu atau beberapa aturan.

```
<ReplicationConfiguration>
```

```
    <Role>IAM-role-ARN</Role>
    <Rule>
        ...
    </Rule>
    <Rule>
         ...
    </Rule>
     ...
</ReplicationConfiguration>
```

Amazon S3 tidak dapat mereplikasi objek tanpa izin Anda. Anda memberikan izin dengan peran IAM yang Anda tentukan dalam konfigurasi replikasi. Amazon S3 mengasumsikan peran IAM untuk mereplikasi objek atas nama Anda. Anda harus terlebih dahulu memberikan izin yang diperlukan untuk peran IAM. Untuk informasi selengkapnya tentang mengelola izin, lihat Menyiapkan izin untuk replikasi langsung.

Anda menambahkan satu aturan dalam konfigurasi replikasi dalam skenario berikut:

- Anda ingin mereplikasi semua objek.
- Anda ingin mereplikasi satu subset objek. Anda mengidentifikasi subset objek dengan menambahkan filter pada aturan. Dalam filter, Anda menentukan awalan kunci objek, tag, atau kombinasi keduanya, untuk mengidentifikasi bagian objek yang diterapkan aturan. Filter menargetkan objek yang sesuai dengan nilai persis yang Anda tentukan.

Anda menambahkan beberapa aturan dalam konfigurasi replikasi jika Anda ingin mereplikasi subset objek yang berbeda. Dalam setiap aturan, Anda menetapkan filter yang memilih subset objek yang berbeda. Misalnya, Anda dapat memilih untuk mereplikasi objek yang memiliki awalan kunci `tax/` atau `document/`. Untuk melakukan ini, Anda menambahkan dua aturan, satu yang menentukan filter awalan kunci `tax/` dan satu lagi yang menentukan awalan kunci `document/`. Untuk informasi selengkapnya tentang tag objek, lihat Organisasi objek menggunakan prefiks.

Bagian berikut memberikan informasi tambahan.

Topik

- Konfigurasi aturan dasar
- Opsional: Menentukan filter
- Konfigurasi tujuan tambahan
- Contoh konfigurasi replikasi

- Kompatibilitas mundur

Konfigurasi aturan dasar

Setiap aturan harus menyertakan status dan prioritas aturan. Aturan juga harus menunjukkan apakah akan mereplikasi penanda hapus.

- `Status` menunjukkan apakah aturan diaktifkan atau dinonaktifkan dengan menggunakan nilai `Enabled` atau `Disabled`. Jika aturan dinonaktifkan, Amazon S3 tidak akan melakukan tindakan yang ditentukan dalam aturan tersebut.
- `Priority` menunjukkan aturan mana yang didahulukan setiap kali dua atau lebih aturan replikasi bertentangan. Amazon S3 mencoba untuk mereplikasi objek sesuai dengan semua aturan replikasi. Namun, jika ada dua aturan atau lebih dengan bucket tujuan yang sama, objek akan direplikasi sesuai aturan dengan prioritas tertinggi. Semakin tinggi angkanya, semakin tinggi prioritasnya.
- `DeleteMarkerReplication` menunjukkan apakah akan mereplikasi penanda hapus dengan menggunakan nilai `Enabled` atau `Disabled`.

Dalam konfigurasi tujuan, Anda harus memberikan nama bucket atau bucket tempat Anda ingin Amazon S3 mereplikasi objek.

Contoh berikut menunjukkan persyaratan minimum untuk aturan V2. Untuk kompatibilitas mundur, Amazon S3 terus mendukung format XML V1. Untuk informasi selengkapnya, lihat Kompatibilitas mundur.

```
...
    <Rule>
        <ID>Rule-1</ID>
        <Status>Enabled-or-Disabled</Status>
        <Filter>
            <Prefix></Prefix>
        </Filter>
        <Priority>integer</Priority>
        <DeleteMarkerReplication>
           <Status>Enabled-or-Disabled</Status>
        </DeleteMarkerReplication>
        <Destination>
           <Bucket>arn:aws:s3:::example-s3-bucket</Bucket>
        </Destination>
```

```
    </Rule>
    <Rule>
         ...
    </Rule>
     ...
...
```

Anda juga dapat menentukan opsi konfigurasi lainnya. Misalnya, Anda dapat memilih untuk menggunakan kelas penyimpanan untuk replika objek yang berbeda dengan kelas untuk objek sumber.

Opsional: Menentukan filter

Untuk memilih subset objek yang menerapkan aturan, tambahkan filter opsional. Anda dapat memfilter berdasarkan awalan kunci objek, tag objek, atau kombinasi keduanya. Jika Anda memfilter kunci prefiks dan tag objek, Amazon S3 menggabungkan filter dengan menggunakan operator `AND` logis. Dengan kata lain, aturan berlaku untuk subset objek dengan awalan kunci khusus dan tag tertentu.

Pemfilteran berdasarkan awalan kunci objek

Untuk menentukan aturan dengan filter berdasarkan awalan kunci objek, gunakan kode berikut. Anda hanya dapat menentukan satu awalan.

```
<Rule>
    ...
    <Filter>
        <Prefix>key-prefix</Prefix>
    </Filter>
    ...
</Rule>
...
```

Pemfilteran berdasarkan tag objek

Untuk menetapkan aturan dengan filter berdasarkan tag objek, gunakan kode berikut. Anda dapat menentukan satu atau beberapa tag objek.

```
<Rule>
    ...
```

```
    <Filter>
        <And>
            <Tag>
                <Key>key1</Key>
                <Value>value1</Value>
            </Tag>
            <Tag>
                <Key>key2</Key>
                <Value>value2</Value>
            </Tag>
             ...
        </And>
    </Filter>
    ...
</Rule>
...
```

Filter dengan prefiks kunci dan tag objek

Untuk menetapkan filter aturan dengan kombinasi awalan kunci dan tag objek, gunakan kode berikut. Anda membungkus filter ini dalam elemen induk <And>. Amazon S3 melakukan operasi AND logis untuk menggabungkan filter ini. Dengan kata lain, aturan berlaku untuk subset objek dengan awalan kunci khusus maupun tag tertentu.

```
<Rule>
    ...
    <Filter>
        <And>
            <Prefix>key-prefix</Prefix>
            <Tag>
                <Key>key1</Key>
                <Value>value1</Value>
            </Tag>
            <Tag>
                <Key>key2</Key>
                <Value>value2</Value>
            </Tag>
             ...
    </Filter>
    ...
</Rule>
...
```

> **Note**
>
> - Jika Anda menentukan aturan dengan `<Filter>` elemen kosong, aturan Anda berlaku untuk semua objek di bucket Anda.
> - Saat Anda menggunakan aturan replikasi berbasis tag dengan replikasi langsung, objek baru harus ditandai dengan tag aturan replikasi yang cocok dalam operasi. `PutObject` Jika tidak, objek tidak akan direplikasi. Jika objek diberi tag setelah `PutObject` operasi, objek tersebut juga tidak akan direplikasi.
>
> Untuk mereplikasi objek yang telah ditandai setelah `PutObject` operasi, Anda harus menggunakan Replikasi Batch S3. Untuk informasi selengkapnya tentang Replikasi Batch, lihat Mereplikasi objek yang ada.

Konfigurasi tujuan tambahan

Dalam konfigurasi tujuan, Anda menentukan bucket tempat Anda ingin Amazon S3 mereplikasi objek. Anda dapat mengatur konfigurasi untuk mereplikasi objek dari satu bucket sumber ke satu atau beberapa bucket tujuan.

```
...
<Destination>
    <Bucket>arn:aws:s3:::example-s3-bucket</Bucket>
</Destination>
...
```

Anda dapat menambahkan opsi berikut di elemen `<Destination>`.

Topik

- Menentukan kelas penyimpanan
- Menambahkan beberapa bucket tujuan
- Menentukan parameter berbeda untuk setiap aturan replikasi dengan beberapa bucket tujuan
- Mengubah kepemilikan replika
- Mengaktifkan Kontrol Waktu Replikasi S3
- Replikasi objek yang dibuat dengan enkripsi sisi server dengan menggunakan AWS KMS

Menentukan kelas penyimpanan

Anda dapat menentukan kelas penyimpanan untuk replika objek. Secara default, Amazon S3 menggunakan kelas penyimpanan objek sumber untuk membuat replika objek, seperti dalam contoh berikut.

```
...
<Destination>
       <Bucket>arn:aws:s3:::example-s3-bucket</Bucket>
       <StorageClass>storage-class</StorageClass>
</Destination>
...
```

Menambahkan beberapa bucket tujuan

Anda dapat menambahkan beberapa bucket tujuan dalam satu konfigurasi replikasi, seperti berikut.

```
...
<Rule>
    <ID>Rule-1</ID>
    <Status>Enabled-or-Disabled</Status>
    <Priority>integer</Priority>
    <DeleteMarkerReplication>
       <Status>Enabled-or-Disabled</Status>
    </DeleteMarkerReplication>
    <Destination>
       <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET1</Bucket>
    </Destination>
</Rule>
<Rule>
    <ID>Rule-2</ID>
    <Status>Enabled-or-Disabled</Status>
    <Priority>integer</Priority>
    <DeleteMarkerReplication>
       <Status>Enabled-or-Disabled</Status>
    </DeleteMarkerReplication>
    <Destination>
       <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET2</Bucket>
    </Destination>
</Rule>
...
```

Menentukan parameter berbeda untuk setiap aturan replikasi dengan beberapa bucket tujuan

Ketika menambahkan beberapa bucket tujuan di satu konfigurasi replikasi, Anda dapat menentukan parameter berbeda untuk setiap aturan replikasi, seperti berikut.

```
...
<Rule>
    <ID>Rule-1</ID>
    <Status>Enabled-or-Disabled</Status>
    <Priority>integer</Priority>
    <DeleteMarkerReplication>
       <Status>Disabled</Status>
    </DeleteMarkerReplication>
      <Metrics>
    <Status>Enabled</Status>
    <EventThreshold>
      <Minutes>15</Minutes>
    </EventThreshold>
  </Metrics>
    <Destination>
       <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET1</Bucket>
    </Destination>
</Rule>
<Rule>
    <ID>Rule-2</ID>
    <Status>Enabled-or-Disabled</Status>
    <Priority>integer</Priority>
    <DeleteMarkerReplication>
       <Status>Enabled</Status>
    </DeleteMarkerReplication>
      <Metrics>
    <Status>Enabled</Status>
    <EventThreshold>
      <Minutes>15</Minutes>
    </EventThreshold>
  </Metrics>
  <ReplicationTime>
    <Status>Enabled</Status>
    <Time>
      <Minutes>15</Minutes>
    </Time>
  </ReplicationTime>
    <Destination>
       <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET2</Bucket>
```

```
    </Destination>
</Rule>
...
```

Mengubah kepemilikan replika

Jika bucket sumber dan tujuan tidak dimiliki oleh akun yang sama, Anda dapat mengubah kepemilikan replika menjadi bucket Akun AWS yang memiliki tujuan. Untuk melakukannya, tambahkan elemen `AccessControlTranslation`. Elemen ini mengambil nilai `Destination`.

```
...
<Destination>
   <Bucket>arn:aws:s3:::example-s3-bucket</Bucket>
   <Account>destination-bucket-owner-account-id</Account>
   <AccessControlTranslation>
       <Owner>Destination</Owner>
   </AccessControlTranslation>
</Destination>
...
```

Jika Anda tidak menambahkan `AccessControlTranslation` elemen ke konfigurasi replikasi, replika dimiliki oleh yang sama Akun AWS yang memiliki objek sumber. Untuk informasi selengkapnya, lihat Mengubah pemilik replika.

Mengaktifkan Kontrol Waktu Replikasi S3

Anda dapat mengaktifkan Kontrol Waktu Replikasi S3 (S3 RTC) dalam konfigurasi replikasi Anda. S3 RTC mereplikasi sebagian besar objek dalam hitungan detik dan 99,99 persen objek dalam waktu 15 menit (didukung oleh perjanjian tingkat layanan).

> **Note**
>
> Hanya nilai `<Minutes>15</Minutes>` yang diterima untuk `EventThreshold` dan `Time`.

```
...
<Destination>
  <Bucket>arn:aws:s3:::example-s3-bucket</Bucket>
  <Metrics>
    <Status>Enabled</Status>
    <EventThreshold>
```

```
      <Minutes>15</Minutes>
    </EventThreshold>
  </Metrics>
  <ReplicationTime>
    <Status>Enabled</Status>
    <Time>
      <Minutes>15</Minutes>
    </Time>
  </ReplicationTime>
</Destination>
...
```

Untuk informasi selengkapnya, lihat Memenuhi persyaratan kepatuhan menggunakan Kontrol Waktu Replikasi S3 (S3 RTC). Untuk contoh API, lihat PutBucketReplicationdi Referensi API Amazon Simple Storage Service.

Replikasi objek yang dibuat dengan enkripsi sisi server dengan menggunakan AWS KMS

Bucket sumber Anda mungkin berisi objek yang dibuat dengan enkripsi sisi server menggunakan kunci AWS Key Management Service (AWS KMS) (SSE-KMS). Secara default, Amazon S3 tidak mereplikasi objek ini. Anda secara opsional dapat mengarahkan Amazon S3 untuk mereplikasi objek ini. Untuk melakukannya, ikut serta dalam fitur ini secara eksplisit terlebih dahulu dengan menambahkan elemen `SourceSelectionCriteria`. Kemudian berikan AWS KMS key (untuk bucket tujuan) yang akan digunakan untuk mengenkripsi replika objek. Wilayah AWS Contoh berikut menunjukkan cara menentukan elemen-elemen ini.

```
...
<SourceSelectionCriteria>
  <SseKmsEncryptedObjects>
    <Status>Enabled</Status>
  </SseKmsEncryptedObjects>
</SourceSelectionCriteria>
<Destination>
  <Bucket>arn:aws:s3:::example-s3-bucket</Bucket>
  <EncryptionConfiguration>
    <ReplicaKmsKeyID>AWS KMS key ID to use for encrypting object replicas</
ReplicaKmsKeyID>
  </EncryptionConfiguration>
</Destination>
...
```

Untuk informasi selengkapnya, lihat Mereplikasi objek terenkripsi (SSE-S3, SSE-KMS, SSE-KMS).

Contoh konfigurasi replikasi

Untuk memulai, Anda dapat menambahkan contoh konfigurasi replikasi berikut ke bucket Anda, sesuai kebutuhan.

> **Important**
>
> Untuk menambahkan konfigurasi replikasi ke bucket, Anda harus memiliki izin `iam:PassRole`. Dengan izin ini, Anda dapat meneruskan peran IAM yang memberikan izin replikasi Amazon S3. Anda menentukan peran IAM dengan menyediakan Amazon Resource Name (ARN) yang digunakan dalam elemen `Role` di XML konfigurasi replikasi. Untuk informasi selengkapnya, lihat Memberikan Izin Pengguna untuk Meneruskan Peran ke Layanan AWS di Panduan Pengguna IAM.

Example 1: Konfigurasi replikasi dengan satu aturan

Konfigurasi replikasi dasar berikut menentukan satu aturan. Aturan tersebut menentukan peran IAM yang dapat diasumsikan Amazon S3 dan satu bucket tujuan untuk replika objek. Nilai `Status` `Enabled` menunjukkan bahwa aturan tersebut berlaku.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::account-id:role/role-name</Role>
  <Rule>
    <Status>Enabled</Status>

    <Destination><Bucket>arn:aws:s3:::example-s3-bucket</Bucket></Destination>

  </Rule>
</ReplicationConfiguration>
```

Untuk memilih subset objek yang akan direplikasi, Anda dapat menambahkan filter. Dalam konfigurasi berikut, filter menentukan awalan kunci objek. Aturan ini berlaku untuk objek yang memiliki awalan *`Tax/`* dalam nama kuncinya.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::account-id:role/role-name</Role>
  <Rule>
```

```
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>

    <Filter>
       <Prefix>Tax/</Prefix>
    </Filter>

    <Destination><Bucket>arn:aws:s3:::example-s3-bucket</Bucket></Destination>

  </Rule>
</ReplicationConfiguration>
```

Jika menentukan elemen `Filter`, Anda juga harus menyertakan elemen `Priority` dan `DeleteMarkerReplication`. Dalam contoh ini, `Priority` tidak relevan karena hanya ada satu aturan.

Dalam konfigurasi berikut, filter menentukan satu awalan dan dua tag. Aturan ini berlaku untuk subset objek yang memiliki awalan dan tag kunci yang ditentukan. Secara khusus, ini berlaku untuk objek yang memiliki awalan *`Tax/`* di nama kunci dan dua tag objek yang ditentukan. `Priority` Tidak berlaku karena hanya ada satu aturan.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::account-id:role/role-name</Role>
  <Rule>
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>

    <Filter>
        <And>
          <Prefix>Tax/</Prefix>
          <Tag>
             <Tag>
                <Key>tagA</Key>
                <Value>valueA</Value>
             </Tag>
          </Tag>
```

```
          <Tag>
             <Tag>
                <Key>tagB</Key>
                <Value>valueB</Value>
             </Tag>
          </Tag>
       </And>

    </Filter>

    <Destination><Bucket>arn:aws:s3:::example-s3-bucket</Bucket></Destination>

  </Rule>
</ReplicationConfiguration>
```

Anda dapat menentukan kelas penyimpanan untuk replika objek sebagai berikut.

```
<?xml version="1.0" encoding="UTF-8"?>

<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::account-id:role/role-name</Role>
  <Rule>
    <Status>Enabled</Status>
    <Destination>
       <Bucket>arn:aws:s3:::example-s3-bucket</Bucket>
       <StorageClass>storage-class</StorageClass>
    </Destination>
  </Rule>
</ReplicationConfiguration>
```

Anda dapat menentukan kelas penyimpanan apa pun yang didukung Amazon S3.

Example 2: Konfigurasi replikasi dengan dua aturan

Example

Dalam konfigurasi replikasi berikut ini:

- Setiap aturan memfilter pada awalan kunci yang berbeda sehingga setiap aturan berlaku untuk subset objek yang berbeda. Dalam contoh ini, Amazon S3 mereplikasi objek dengan nama kunci *Tax/doc1.pdf* dan *Project/project1.txt*, tetapi tidak mereplikasi objek dengan nama kunci *PersonalDoc/documentA*.

- Prioritas aturan tidak relevan karena aturan berlaku pada dua set objek yang berbeda. Contoh berikutnya menunjukkan apa yang terjadi ketika prioritas aturan diterapkan.
- Aturan kedua menetapkan kelas penyimpanan S3 Standard-IA untuk replika objek. Amazon S3 menggunakan kelas penyimpanan yang ditentukan untuk replika objek tersebut.

```
<?xml version="1.0" encoding="UTF-8"?>

<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Role>arn:aws:iam::account-id:role/role-name</Role>
  <Rule>
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>
    <Filter>
        <Prefix>Tax</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Destination>
      <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET1</Bucket>
    </Destination>
     ...
  </Rule>
 <Rule>
    <Status>Enabled</Status>
    <Priority>2</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>
    <Filter>
        <Prefix>Project</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Destination>
      <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET1</Bucket>
     <StorageClass>STANDARD_IA</StorageClass>
    </Destination>
     ...
  </Rule>
```

```
</ReplicationConfiguration>
```

Example 3: Konfigurasi replikasi dengan dua aturan dengan awalan yang tumpang tindih

Dalam konfigurasi ini, dua aturan yang menentukan filter dengan awalan kunci yang tumpang tindih, *star/* dan *starship/*. Kedua aturan berlaku untuk objek dengan nama kunci *starship-x*. Dalam kasus ini, Amazon S3 menggunakan aturan prioritas untuk menentukan aturan mana yang diterapkan. Semakin tinggi angkanya, semakin tinggi prioritasnya.

```
<ReplicationConfiguration>

  <Role>arn:aws:iam::account-id:role/role-name</Role>

  <Rule>
    <Status>Enabled</Status>
    <Priority>1</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>
    <Filter>
        <Prefix>star</Prefix>
    </Filter>
    <Destination>
      <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET1</Bucket>
    </Destination>
  </Rule>
  <Rule>
    <Status>Enabled</Status>
    <Priority>2</Priority>
    <DeleteMarkerReplication>
       <Status>string</Status>
    </DeleteMarkerReplication>
    <Filter>
        <Prefix>starship</Prefix>
    </Filter>
    <Destination>
      <Bucket>arn:aws:s3:::DOC-EXAMPLE-BUCKET1</Bucket>
    </Destination>
  </Rule>
</ReplicationConfiguration>
```

Example 4: Contoh panduan

Untuk contoh panduan, lihat Contoh untuk mengonfigurasi replikasi langsung.

Untuk informasi selengkapnya tentang struktur XHTML konfigurasi replikasi, lihat PutBucketReplicationdi Referensi API Amazon Simple Storage Service.

Kompatibilitas mundur

Versi terbaru konfigurasi replikasi XML adalah V2. Konfigurasi replikasi XML V2 adalah yang berisi elemen `Filter` untuk aturan, dan aturan yang menentukan Kontrol Waktu Replikasi S3 (S3 RTC).

Untuk melihat versi konfigurasi replikasi, Anda dapat menggunakan operasi API `GetBucketReplication`. Untuk informasi selengkapnya, lihat GetBucketReplicationdi Referensi API Amazon Simple Storage Service.

Untuk kompatibilitas mundur, Amazon S3 terus mendukung konfigurasi replikasi XML V1. Jika Anda telah menggunakan konfigurasi replikasi XML V1, pertimbangkan masalah berikut yang memengaruhi kompatibilitas mundur:

- Konfigurasi replikasi XML V2 meliputi elemen `Filter` untuk aturan. Dengan elemen `Filter`, Anda dapat menentukan filter objek berdasarkan awalan kunci objek, tag, atau keduanya untuk mencakup objek yang menerapkan aturan tersebut. Konfigurasi replikasi XML V1 mendukung pemfilteran berdasarkan awalan kunci saja. Dalam hal ini, Anda menambahkan `Prefix` secara langsung sebagai elemen anak dari elemen `Rule`, sebagaimana pada contoh berikut.

  ```
  <?xml version="1.0" encoding="UTF-8"?>
  <ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    <Role>arn:aws:iam::account-id:role/role-name</Role>
    <Rule>
      <Status>Enabled</Status>
      <Prefix>key-prefix</Prefix>
      <Destination><Bucket>arn:aws:s3:::example-s3-bucket</Bucket></Destination>

    </Rule>
  </ReplicationConfiguration>
  ```

  Untuk kompatibilitas mundur, Amazon S3 terus mendukung konfigurasi V1.

- Saat Anda menghapus objek dari bucket sumber tanpa menentukan ID versi objek, Amazon S3 menambahkan penanda hapus. Jika Anda menggunakan V1 dari konfigurasi replikasi XML,

Amazon S3 mereplikasi penanda hapus yang muncul karena tindakan pengguna. Dengan kata lain, Amazon S3 mereplikasi penanda hapus hanya jika pengguna menghapus objek. Jika objek kedaluwarsa dihapus oleh Amazon S3 (sebagai bagian dari tindakan siklus hidup), Amazon S3 tidak mereplikasi penanda hapus.

Dalam konfigurasi replikasi V2, Anda dapat mengaktifkan replikasi penanda hapus untuk aturan. non-tag-based Untuk informasi selengkapnya, lihat Mereplikasi penanda hapus di antara bucket.

Menyiapkan izin untuk replikasi langsung

Saat mengatur replikasi langsung, Anda harus memperoleh izin yang diperlukan sebagai berikut:

- Amazon S3 membutuhkan izin untuk mereplikasi objek atas nama Anda. Anda memberikan izin ini dengan membuat peran IAM lalu menentukan peran tersebut dalam konfigurasi replikasi Anda.
- Saat bucket sumber dan tujuan tidak dimiliki oleh akun yang sama, pemilik bucket tujuan harus memberi pemilik bucket sumber izin untuk menyimpan replika.

Topik

- Membuat peran IAM
- Memberikan izin saat bucket sumber dan tujuan dimiliki oleh yang berbeda Akun AWS
- Memberikan izin untuk Operasi Batch S3
- Mengubah kepemilikan replika
- Mengaktifkan penerimaan objek yang direplikasi dari bucket sumber

Membuat peran IAM

Secara default, semua sumber daya Amazon S3—bucket, objek, dan subsumber terkait—bersifat privat dan hanya pemilik sumber daya yang bisa mengakses sumber daya. Amazon S3 membutuhkan izin untuk membaca dan mereplikasi objek dari bucket sumber. Anda memberikan izin ini dengan membuat peran IAM dan menentukan peran tersebut dalam konfigurasi replikasi Anda.

Bagian ini menjelaskan kebijakan kepercayaan dan kebijakan izin minimum yang diperlukan. Contoh penelusuran memberikan step-by-step instruksi untuk membuat peran IAM. Untuk informasi selengkapnya, lihat Contoh untuk mengonfigurasi replikasi langsung.

- Contoh berikut menunjukkan kebijakan kepercayaan, yaitu saat Anda mengidentifikasi Amazon S3 sebagai pengguna utama layanan yang dapat mengasumsikan peran tersebut.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

- Contoh berikut menunjukkan kebijakan kepercayaan, yaitu saat Anda mengidentifikasi Amazon S3 dan Operasi Batch S3 sebagai pengguna utama layanan. Ini berguna jika Anda membuat tugas Replikasi Batch. Untuk informasi selengkapnya, lihat Membuat tugas Replikasi Batch untuk aturan replikasi pertama atau tujuan baru.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service": [
              "s3.amazonaws.com",
              "batchoperations.s3.amazonaws.com"
            ]
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

Untuk informasi selengkapnya tentang peran IAM, lihat Peran IAM dalam Panduan Pengguna IAM.

- Contoh berikut menunjukkan kebijakan akses, yaitu ketika Anda memberikan izin peran untuk melakukan tugas replikasi atas nama Anda. Saat Amazon S3 memegang peran tersebut, Amazon

S3 memiliki izin yang Anda tentukan dalam kebijakan ini. Dalam kebijakan ini, *example-s3-bucket1* adalah bucket sumber, dan *example-s3-bucket2* merupakan bucket tujuan.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetReplicationConfiguration",
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::example-s3-bucket1"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl",
            "s3:GetObjectVersionTagging"
         ],
         "Resource":[
            "arn:aws:s3:::example-s3-bucket1/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ReplicateObject",
            "s3:ReplicateDelete",
            "s3:ReplicateTags"
         ],
         "Resource":"arn:aws:s3:::example-s3-bucket2/*"
      }
   ]
}
```

Kebijakan akses memberikan izin untuk tindakan berikut:

- `s3:GetReplicationConfiguration` dan `s3:ListBucket`–Izin untuk tindakan ini di bucket *example-s3-bucket1* (bucket sumber) memungkinkan Amazon S3 untuk mengambil

konfigurasi replikasi dan mencantumkan konten bucket. (Model izin saat ini memerlukan izin `s3:ListBucket` untuk mengakses penanda hapus.)

- `s3:GetObjectVersionForReplication` dan `s3:GetObjectVersionAcl`–Izin untuk tindakan ini diberikan pada semua objek untuk memungkinkan Amazon S3 mendapatkan versi objek tertentu dan daftar kontrol akses (ACL) yang terkait dengan objek.

- `s3:ReplicateObject` dan `s3:ReplicateDelete`–Izin untuk tindakan ini pada semua objek di bucket *`example-s3-bucket2`* (bucket tujuan) memungkinkan Amazon S3 mereplikasi objek atau penanda hapus ke bucket tujuan. Untuk informasi tentang penanda hapus, lihat Bagaimana cara menghapus operasi yang memengaruhi replikasi.

  > **Note**
  >
  > Izin untuk tindakan `s3:ReplicateObject` pada bucket *`example-s3-bucket2`* (bucket tujuan) juga memungkinkan replikasi metadata seperti tag objek dan ACLS. Oleh karena itu Anda tidak perlu secara eksplisit memberikan izin untuk tindakan `s3:ReplicateTags` tersebut.

- `s3:GetObjectVersionTagging`–Izin untuk tindakan ini pada objek di dalam bucket *`example-s3-bucket1`* (bucket sumber) memungkinkan Amazon S3 membaca tag objek untuk replikasi. Untuk informasi selengkapnya, lihat Mengategorikan penyimpanan Anda menggunakan tag. Jika Amazon S3 tidak memiliki izin ini, Amazon S3 akan mereplikasi objek, tetapi tidak dengan tag objek.

Untuk daftar tindakan Amazon S3, lihat Tindakan, sumber daya, dan kunci kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

> **Important**
>
> Akun AWS Yang memiliki peran IAM harus memiliki izin untuk tindakan yang diberikannya ke peran IAM.
> Misalnya, anggaplah bucket sumber berisi objek yang dimiliki oleh Akun AWS lain. Pemilik objek harus secara eksplisit memberikan Akun AWS yang memiliki peran IAM izin yang diperlukan melalui objek ACL. Jika tidak, Amazon S3 tidak dapat mengakses objek, dan replikasi objek akan gagal. Untuk informasi tentang izin ACL, lihat Gambaran umum daftar kontrol akses (ACL).

Izin yang dijelaskan di sini terkait dengan konfigurasi replikasi minimum. Jika Anda memilih untuk menambahkan konfigurasi replikasi opsional, Anda harus memberikan izin tambahan ke Amazon S3.

Memberikan izin saat bucket sumber dan tujuan dimiliki oleh yang berbeda Akun AWS

Saat bucket sumber dan tujuan tidak dimiliki oleh akun yang sama, pemilik bucket tujuan juga harus menambahkan kebijakan bucket guna memberi pemilik bucket sumber izin untuk melakukan tindakan replikasi sebagai berikut. Dalam kebijakan ini, *example-s3-bucket2* adalah bucket tujuan.

> **Note**
>
> Format ARN peran mungkin tampak berbeda. Jika peran dibuat dengan menggunakan konsol, format ARN adalah. `arn:aws:iam::account-ID:role/service-role/role-name` Jika peran dibuat dengan menggunakan AWS CLI, format ARN adalah. `arn:aws:iam::account-ID:role/role-name` Untuk informasi selengkapnya, lihat peran IAM dalam Panduan Pengguna IAM.

```
{
   "Version":"2012-10-17",
   "Id":"PolicyForDestinationBucket",
   "Statement":[
      {
         "Sid":"Permissions on objects",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::SourceBucket-account-ID:role/service-role/source-
account-IAM-role"
         },
         "Action":[
            "s3:ReplicateDelete",
            "s3:ReplicateObject"
         ],
         "Resource":"arn:aws:s3:::example-s3-bucket2/*"
      },
      {
         "Sid":"Permissions on bucket",
         "Effect":"Allow",
```

```
         "Principal":{
            "AWS":"arn:aws:iam::SourceBucket-account-ID:role/service-role/source-
account-IAM-role"
         },
         "Action": [
            "s3:List*",
            "s3:GetBucketVersioning",
            "s3:PutBucketVersioning"
         ],
         "Resource":"arn:aws:s3::::example-s3-bucket2"
      }
   ]
}
```

Sebagai contoh, lihat Mengonfigurasi replikasi ketika bucket sumber dan tujuan dimiliki oleh akun yang berbeda.

Jika objek dalam bucket sumber ditandai, perhatikan hal berikut:

- Jika pemilik bucket sumber memberikan izin kepada Amazon S3 untuk tindakan `s3:GetObjectVersionTagging` dan `s3:ReplicateTags` guna mereplikasi tag objek (melalui peran IAM), Amazon S3 mereplikasi tag beserta objek. Untuk informasi tentang peran IAM, lihat Membuat peran IAM.
- Jika pemilik bucket tujuan tidak ingin mereplikasi tag, mereka dapat menambahkan pernyataan berikut ke kebijakan bucket tujuan untuk secara eksplisit menolak izin bagi tindakan `s3:ReplicateTags`. Dalam kebijakan ini, *`example-s3-bucket2`* adalah bucket tujuan.

```
...
   "Statement":[
      {
         "Effect":"Deny",
         "Principal":{
            "AWS":"arn:aws:iam::SourceBucket-account-id:role/service-role/source-
account-IAM-role"
         },
         "Action":"s3:ReplicateTags",
         "Resource":"arn:aws:s3:::example-s3-bucket2/*"
      }
   ]
...
```

Memberikan izin untuk Operasi Batch S3

Replikasi Batch S3 memberi Anda cara untuk mereplikasi objek yang ada sebelum konfigurasi replikasi ada, objek yang sebelumnya telah direplikasi, dan objek yang gagal direplikasi. Anda dapat membuat tugas Replikasi Batch satu kali saat membuat aturan pertama dalam konfigurasi replikasi baru atau saat menambahkan tujuan baru ke konfigurasi yang ada melalui AWS Management Console. Anda juga dapat memulai Replikasi Batch untuk konfigurasi replikasi yang ada dengan membuat tugas Operasi Batch.

Untuk contoh peran dan kebijakan IAM Replikasi Batch, lihat, Mengonfigurasi kebijakan IAM untuk Replikasi Batch.

Mengubah kepemilikan replika

Jika berbeda Akun AWS memiliki bucket sumber dan tujuan, Anda dapat memberi tahu Amazon S3 untuk mengubah kepemilikan replika ke bucket Akun AWS yang memiliki tujuan. Untuk informasi selengkapnya tentang penggantian pemilik, lihat Mengubah pemilik replika.

Mengaktifkan penerimaan objek yang direplikasi dari bucket sumber

Anda dapat dengan cepat membuat kebijakan yang diperlukan untuk mengaktifkan penerimaan objek yang direplikasi dari bucket sumber melalui AWS Management Console.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di daftar Bucket, pilih bucket yang ingin Anda gunakan sebagai bucket tujuan.
4. Pilih tab Manajemen, lalu gulir ke bawah ke Aturan replikasi.
5. Untuk Tindakan, pilih Menerima objek yang direplikasi.

   Ikuti petunjuknya dan masukkan Akun AWS ID akun bucket sumber, lalu pilih Buat kebijakan. Ini akan menghasilkan kebijakan bucket Amazon S3 dan kebijakan kunci KMS.
6. Untuk menambahkan kebijakan ini ke kebijakan bucket yang ada, pilih Terapkan pengaturan atau pilih Salin untuk menyalin perubahan secara manual.
7. (Opsional) Salin AWS KMS kebijakan ke kebijakan kunci KMS yang Anda inginkan di AWS Key Management Service konsol.

Contoh untuk mengonfigurasi replikasi langsung

Contoh-contoh berikut menunjukkan cara mengonfigurasi replikasi langsung untuk kasus penggunaan umum.

> **Note**
>
> Replikasi langsung mengacu pada Replikasi Wilayah yang Sama (SRR) dan Replikasi Lintas-Wilayah (CRR). Replikasi langsung tidak mereplikasi objek apa pun yang ada di bucket sebelum Anda mengatur replikasi. Untuk mereplikasi objek yang ada sebelum Anda mengatur replikasi, gunakan replikasi sesuai permintaan. Untuk menyinkronkan bucket dan mereplikasi objek yang ada sesuai permintaan, lihat. Mereplikasi objek yang ada

Contoh-contoh ini menunjukkan cara membuat konfigurasi replikasi dengan menggunakan konsol Amazon S3 AWS Command Line Interface ,AWS CLI(), AWS dan SDK AWS SDK for Java (AWS SDK for .NET dan contoh ditampilkan).

Untuk informasi tentang menginstal dan mengonfigurasi AWS CLI, lihat topik berikut di Panduan AWS Command Line Interface Pengguna.

- Instalasi AWS Command Line Interface
- Mengkonfigurasi AWS CLI — Anda harus mengatur setidaknya satu profil. Jika Anda menjelajahi skenario lintas akun, siapkan dua profil.

Untuk informasi tentang SDK, lihat AWS SDK for Java AWS dan AWS SDK for .NET.

> **Tip**
>
> Untuk step-by-step tutorial yang menunjukkan cara menggunakan replikasi langsung untuk mereplikasi data, lihat Tutorial: Mereplikasi data di dalam dan di antara Wilayah AWS menggunakan Replikasi S3.

Topik

- Mengonfigurasi replikasi untuk bucket sumber dan tujuan yang dimiliki oleh akun yang sama
- Mengonfigurasi replikasi ketika bucket sumber dan tujuan dimiliki oleh akun yang berbeda

- Memenuhi persyaratan kepatuhan menggunakan Kontrol Waktu Replikasi S3 (S3 RTC)
- Mereplikasi objek terenkripsi (SSE-S3, SSE-KMS, SSE-KMS)
- Mereplikasi perubahan metadata dengan sinkronisasi modifikasi replika Amazon S3
- Mereplikasi penanda hapus di antara bucket

Mengonfigurasi replikasi untuk bucket sumber dan tujuan yang dimiliki oleh akun yang sama

Replikasi adalah penyalinan objek secara otomatis dan asinkron di seluruh ember dalam hal yang sama atau berbeda. Wilayah AWS Replikasi menyalin objek yang baru dibuat dan pembaruan objek dari bucket sumber ke bucket atau bucket tujuan. Untuk informasi selengkapnya, lihat Mereplikasi ikhtisar objek.

Saat mengonfigurasi replikasi, Anda menambahkan aturan replikasi ke bucket sumber. Aturan replikasi menentukan objek bucket sumber mana yang akan direplikasi dan bucket tujuan atau bucket tempat objek yang direplikasi disimpan. Anda dapat membuat aturan untuk mereplikasi semua objek dalam bucket atau subset objek dengan awalan nama kunci tertentu, satu atau beberapa tag objek, atau keduanya. Bucket tujuan bisa Akun AWS sama dengan bucket sumber, atau bisa juga di akun yang berbeda.

Jika Anda menentukan ID versi objek yang akan dihapus, Amazon S3 menghapus versi objek tersebut dalam bucket sumber. Tetapi ini tidak mereplikasi penghapusan di bucket tujuan. Dengan kata lain, itu tidak menghapus versi objek yang sama dari bucket tujuan. Ini melindungi data dari penghapusan berbahaya.

Ketika Anda menambahkan aturan replikasi ke bucket, aturan diaktifkan secara default, sehingga aturan mulai bekerja segera setelah Anda menyimpannya.

Dalam contoh ini, Anda menyiapkan replikasi untuk bucket sumber dan tujuan yang dimiliki oleh Akun AWS yang sama. Contoh disediakan untuk menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), dan dan. AWS SDK for Java AWS SDK for .NET

Menggunakan konsol S3

Untuk mengonfigurasi aturan replikasi saat bucket tujuan Akun AWS sama dengan bucket sumber, ikuti langkah-langkah berikut.

Jika bucket tujuan berada di akun yang berbeda dengan bucket sumber, Anda harus menambahkan kebijakan bucket ke bucket tujuan untuk memberi pemilik akun bucket sumber izin untuk mereplikasi

objek dalam bucket tujuan. Untuk informasi selengkapnya, lihat Memberikan izin saat bucket sumber dan tujuan dimiliki oleh yang berbeda Akun AWS.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dari daftar Bucket, pilih nama bucket yang Anda inginkan.
4. Pilih tab Manajemen, gulir ke bawah ke Aturan replikasi, lalu pilih Buat aturan replikasi.
5. Di bagian Konfigurasi aturan replikasi, di bawah Nama aturan replikasi, masukkan nama untuk aturan Anda guna membantu mengidentifikasi aturan tersebut nanti. Nama wajib diisi dan harus unik dalam bucket.
6. Di bawah Status, Diaktifkan dipilih secara default. Aturan yang diaktifkan mulai berfungsi segera setelah Anda menyimpannya. Jika Anda ingin mengaktifkan aturan nanti, pilih Dinonaktifkan.
7. Jika bucket memiliki aturan replikasi yang ada, Anda diinstruksikan untuk menetapkan prioritas aturan. Anda harus menetapkan prioritas pada aturan untuk menghindari konflik yang disebabkan oleh objek yang dimasukkan dalam cakupan lebih dari satu aturan. Dalam kasus aturan yang tumpang tindih, Amazon S3 menggunakan prioritas aturan untuk menentukan aturan mana yang berlaku. Semakin tinggi angkanya, semakin tinggi prioritasnya. Untuk informasi selengkapnya tentang prioritas aturan, lihat Konfigurasi Replikasi.
8. Di bawah Bucket sumber, Anda memiliki opsi berikut untuk mengatur sumber replikasi:
   - Untuk mereplikasi seluruh bucket, pilih Terapkan ke semua objek di bucket.
   - Untuk mereplikasi semua objek yang memiliki awalan yang sama, pilih Batasi cakupan aturan ini menggunakan satu atau beberapa filter. Ini membatasi replikasi ke semua objek yang memiliki nama yang dimulai dengan awalan yang Anda tentukan (misalnya `pictures`). Masukkan awalan di kotak Awalan.

     > **Note**
     >
     > Jika Anda memasukkan awalan yang merupakan nama folder, Anda harus menggunakan / (garis miring ke depan) sebagai karakter terakhir (misalnya, `pictures/`).

   - Untuk mereplikasi semua objek dengan satu atau beberapa tag objek, pilih Tambahkan tag dan masukkan pasangan nilai kunci ke dalam kotak. Ulangi prosedur untuk menambahkan tag

lainnya. Anda dapat menggabungkan awalan dan tag. Untuk informasi selengkapnya tentang tag objek, lihat Mengategorikan penyimpanan Anda menggunakan tag.

Skema XML konfigurasi replikasi baru mendukung pemfilteran awalan dan tag serta prioritas aturan. Untuk informasi selengkapnya tentang skema baru, lihat Kompatibilitas mundur. Untuk informasi selengkapnya tentang XML yang digunakan dengan API Amazon S3 yang berfungsi di balik antarmuka pengguna, lihat Konfigurasi Replikasi. Skema baru dijelaskan sebagai konfigurasi replikasi XML V2.

9. Di bawah Tujuan, pilih bucket tempat Anda ingin Amazon S3 mereplikasi objek.

> **Note**
>
> Jumlah ember tujuan terbatas pada jumlah Wilayah AWS di partisi tertentu. Partisi adalah pengelompokan Wilayah. AWS Saat ini memiliki tiga partisi: `aws` (Wilayah Standar), `aws-cn` (Wilayah China), dan `aws-us-gov` (AWS GovCloud (US) Wilayah). Untuk meminta peningkatan kuota bucket tujuan Anda, Anda dapat menggunakan kuota layanan.

- Untuk mereplikasi ke bucket atau beberapa bucket di akun Anda, pilih Pilih bucket di akun ini, dan masukkan atau telusuri nama bucket tujuan.
- Untuk mereplikasi ke bucket atau bucket yang berbeda Akun AWS, pilih Tentukan bucket di akun lain, lalu masukkan ID akun bucket tujuan dan nama bucket.

  Jika tujuan berada di akun yang berbeda dengan bucket sumber, Anda harus menambahkan kebijakan bucket ke bucket tujuan untuk memberi pemilik akun bucket sumber izin untuk mereplikasi objek. Untuk informasi selengkapnya, lihat Memberikan izin saat bucket sumber dan tujuan dimiliki oleh yang berbeda Akun AWS.

  Secara opsional, jika Anda ingin membantu menstandarkan kepemilikan objek baru di bucket tujuan, pilih Ubah kepemilikan objek ke pemilik bucket tujuan. Untuk informasi selengkapnya tentang metrik ini, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

> **Note**
>
> Jika Penentuan Versi tidak diaktifkan pada bucket tujuan, Anda mendapatkan peringatan yang berisi tombol Aktifkan Penentuan Versi. Pilih tombol ini untuk mengaktifkan versi pada bucket.

10. Siapkan peran AWS Identity and Access Management (IAM) yang dapat diasumsikan Amazon S3 untuk mereplikasi objek atas nama Anda.

    Untuk menyiapkan peran IAM, di bagian peran IAM, pilih salah satu dari daftar dropdown peran IAM berikut ini:

    - Kami sangat merekomendasikan Anda untuk memilih Buat peran baru agar Amazon S3 membuat peran IAM baru untuk Anda. Saat Anda menyimpan aturan, kebijakan baru akan dibuat untuk peran IAM yang sesuai dengan bucket sumber dan tujuan yang Anda pilih.
    - Anda dapat memilih untuk menggunakan peran IAM yang sudah ada. Jika melakukannya, Anda harus memilih peran yang memberi Amazon S3 izin yang diperlukan untuk replikasi. Replikasi akan gagal jika peran ini tidak memberi Amazon S3 izin yang memadai untuk mengikuti aturan replikasi Anda.

    > **Important**
    >
    > Saat menambahkan aturan replikasi ke bucket, Anda harus memiliki izin `iam:PassRole` untuk dapat meneruskan peran IAM yang memberi izin Replikasi Amazon S3. Untuk informasi selengkapnya, lihat Memberikan izin pengguna untuk meneruskan peran ke Layanan AWS di Panduan Pengguna IAM.

11. Untuk mereplikasi objek di bucket sumber yang dienkripsi dengan enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS), di bawah Enkripsi, pilih Replikasi objek yang dienkripsi dengan. AWS KMS Di bawah AWS KMS kunci untuk mengenkripsi objek tujuan adalah kunci sumber yang dapat digunakan replikasi. Semua kunci sumber KMS disertakan secara default. Untuk mempersempit pilihan kunci KMS, Anda dapat memilih alias atau ID kunci.

    Objek yang dienkripsi oleh AWS KMS keys yang tidak Anda pilih tidak direplikasi. Kunci KMS atau kumpulan kunci KMS akan dipilih untuk Anda, tetapi Anda dapat memilih kunci KMS jika

mau. Untuk informasi tentang menggunakan AWS KMS dengan replikasi, lihatMereplikasi objek terenkripsi (SSE-S3, SSE-KMS, SSE-KMS).

> **Important**
>
> Saat Anda mereplikasi objek yang dienkripsi AWS KMS, tingkat AWS KMS permintaan berlipat ganda di Wilayah sumber dan meningkat di Wilayah tujuan dengan jumlah yang sama. Tingkat panggilan yang meningkat ini disebabkan oleh cara data dienkripsi ulang dengan menggunakan kunci KMS yang Anda tentukan untuk Wilayah tujuan replikasi. AWS KMS AWS KMS memiliki kuota tarif permintaan per rekening panggilan per Region. Untuk informasi tentang default kuota, lihat Kuota AWS KMS -Permintaan per Detik: Bervariasi di Panduan Developer AWS Key Management Service .
>
> Jika tingkat permintaan PUT objek Amazon S3 saat ini selama replikasi lebih dari setengah batas AWS KMS tarif default untuk akun Anda, kami sarankan Anda meminta kenaikan kuota tingkat AWS KMS permintaan Anda. Untuk meminta peningkatan, buat kasus di Pusat AWS Support di Hubungi Kami. Misalnya, misalkan tingkat permintaan PUT objek Anda saat ini adalah 1.000 permintaan per detik dan Anda gunakan AWS KMS untuk mengenkripsi objek Anda. Dalam hal ini, kami menyarankan agar Anda meminta AWS Support untuk meningkatkan batas AWS KMS tarif Anda menjadi 2.500 permintaan per detik, baik di Wilayah sumber maupun tujuan Anda (jika berbeda), untuk memastikan bahwa tidak ada pembatasan oleh. AWS KMS
>
> Untuk melihat rasio permintaan PUT objek Anda di bucket sumber, lihat `PutRequests` di metrik CloudWatch permintaan Amazon untuk Amazon S3. Untuk informasi tentang melihat CloudWatch metrik, lihatMenggunakan konsol S3.

Jika Anda memilih untuk mereplikasi objek yang dienkripsi AWS KMS, lakukan hal berikut:

- Di bawah AWS KMS key untuk mengenkripsi objek tujuan, tentukan kunci KMS Anda dengan salah satu cara berikut:
  - Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari AWS KMS keys, dan pilih Kunci KMS Anda dari daftar kunci yang tersedia.

    Kunci Kunci yang dikelola AWS (`aws/s3`) dan kunci terkelola pelanggan Anda muncul dalam daftar ini. Untuk informasi selengkapnya tentang CMK, lihat Kunci pelanggan dan AWS kunci di AWS Key Management Service Panduan Pengembang.

- Untuk memasukkan kunci KMS Amazon Resource Name (ARN), pilih Masukkan ARN AWS KMS key , dan masukkan ARN kunci KMS Anda di bidang yang muncul. Ini akan mengenkripsi replika di bucket tujuan. Anda dapat menemukan ARN untuk kunci KMS Anda di Konsol IAM, di bawah kunci Enkripsi.
- Untuk membuat kunci terkelola pelanggan baru di AWS KMS konsol, pilih Buat kunci KMS.

  Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat kunci di Panduan AWS Key Management Service Pengembang.

> **Important**
>
> Anda hanya dapat menggunakan tombol KMS yang diaktifkan Wilayah AWS sama dengan bucket. Saat memilih Pilih dari kunci KMS Anda, konsol S3 hanya mencantumkan 100 kunci KMS per Wilayah. Jika Anda memiliki lebih dari 100 tombol KMS di Wilayah yang sama, Anda hanya dapat melihat 100 kunci KMS pertama di konsol S3. Untuk menggunakan kunci KMS yang tidak tercantum di konsol, pilih Masukkan ARN AWS KMS key , lalu masukkan ARN kunci KMS Anda. Saat Anda menggunakan enkripsi sisi server AWS KMS key untuk Amazon S3, Anda harus memilih kunci KMS enkripsi simetris. Amazon S3 hanya mendukung kunci KMS enkripsi simetris dan tidak mendukung kunci KMS asimetris. Untuk informasi selengkapnya, lihat Mengidentifikasi tombol KMS simetris dan asimetris dalam Panduan Pengembang AWS Key Management Service .

Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat kunci di Panduan AWS Key Management Service Pengembang. Untuk informasi selengkapnya tentang penggunaan AWS KMS dengan Amazon S3, lihat. Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS)

12. Di Bawah Kelas penyimpanan tujuan, jika Anda ingin mereplikasi data Anda ke dalam kelas penyimpanan tertentu di tujuan, pilih Ubah kelas penyimpanan untuk objek yang direplikasi. Kemudian pilih kelas penyimpanan yang ingin Anda gunakan untuk objek yang direplikasi di tujuan. Jika Anda tidak memilih opsi ini, kelas penyimpanan untuk objek yang direplikasi adalah kelas yang sama dengan objek aslinya.
13. Anda memiliki opsi tambahan berikut saat mengatur Opsi replikasi tambahan:

- Jika Anda ingin mengaktifkan Kontrol Waktu Replikasi S3 (S3 RTC) dalam konfigurasi replikasi Anda, pilih Kontrol Waktu Replikasi (RTC). Untuk informasi selengkapnya tentang metrik ini, lihat Memenuhi persyaratan kepatuhan menggunakan Kontrol Waktu Replikasi S3 (S3 RTC).
- Jika Anda ingin mengaktifkan metrik replikasi S3 dalam konfigurasi replikasi Anda, pilih Metrik dan peristiwa replikasi. Untuk informasi selengkapnya, lihat, Memantau progres dengan metrik replikasi dan Notifikasi Peristiwa S3.
- Jika Anda ingin mengaktifkan replikasi penanda hapus dalam konfigurasi replikasi Anda, pilih Replikasi penanda hapus. Untuk informasi selengkapnya, lihat, Mereplikasi penanda hapus di antara bucket.
- Jika Anda ingin mengaktifkan sinkronisasi modifikasi replika Amazon S3 dalam konfigurasi replikasi Anda, pilih Sinkronisasi modifikasi replika. Untuk informasi selengkapnya, lihat, Mereplikasi perubahan metadata dengan sinkronisasi modifikasi replika Amazon S3.

> **Note**
>
> Ketika Anda menggunakan metrik Replikasi S3 atau S3 RTC, biaya tambahan berlaku.

14. Untuk menyelesaikan, pilih Simpan.
15. Setelah menyimpan aturan, Anda dapat mengedit, mengaktifkan, menonaktifkan, atau menghapus aturan Anda dengan memilih aturan dan memilih Edit aturan.

Menggunakan AWS CLI

Untuk menggunakan AWS CLI untuk mengatur replikasi ketika bucket sumber dan tujuan dimiliki oleh yang sama Akun AWS, Anda melakukan hal berikut:

- Membuat bucket sumber dan tujuan
- Mengaktifkan Penentuan Versi pada bucket
- Membuat peran IAM yang memberikan izin Amazon S3 untuk mereplikasi objek
- Menambahkan konfigurasi replikasi ke bucket sumber

Untuk memverifikasi konfigurasi, Anda mengujinya.

Untuk mengatur replikasi saat bucket sumber dan tujuan dimiliki oleh yang sama Akun AWS

1. Atur profil kredensial untuk AWS CLI. Dalam contoh ini, kami menggunakan nama profil `acctA`. Untuk informasi tentang mengatur profil kredensial, lihat Profil Bernama di Panduan Pengguna AWS Command Line Interface .

   > **Important**
   >
   > Profil yang Anda gunakan untuk latihan ini harus memiliki izin yang diperlukan. Misalnya, dalam konfigurasi replikasi, Anda menentukan peran IAM yang dapat diasumsikan oleh Amazon S3. Anda dapat melakukan ini hanya jika profil yang Anda gunakan memiliki izin `iam:PassRole`. Untuk informasi selengkapnya, lihat Memberikan Izin Pengguna untuk Meneruskan Peran ke Layanan AWS di Panduan Pengguna IAM. Jika Anda menggunakan kredensial administrator untuk membuat profil bernama, Anda dapat melakukan semua tugas.

2. Buat *`source`* bucket dan aktifkan penentuan versi di dalamnya. Kode berikut membuat bucket *`source`* di Wilayah AS Timur (Virginia Utara) (us-east-1).

```
aws s3api create-bucket \
--bucket source \
--region us-east-1 \
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket source \
--versioning-configuration Status=Enabled \
--profile acctA
```

3. Buat *`destination`* bucket dan aktifkan penentuan versi di dalamnya. Kode berikut membuat bucket *`destination`* di Wilayah AS Barat (Oregon) (us-west-2).

   > **Note**
   >
   > Untuk mengatur konfigurasi replikasi saat bucket sumber dan tujuan berada di tempat yang sama Akun AWS, Anda menggunakan profil yang sama. Contoh ini menggunakan `acctA`. Untuk menguji konfigurasi replikasi ketika bucket dimiliki oleh yang berbeda

Akun AWS, Anda menentukan profil yang berbeda untuk masing-masing. Contoh ini menggunakan profil `acctB` untuk bucket tujuan.

```
aws s3api create-bucket \
--bucket destination \
--region us-west-2 \
--create-bucket-configuration LocationConstraint=us-west-2 \
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket destination \
--versioning-configuration Status=Enabled \
--profile acctA
```

4. Buat peran IAM. Anda menentukan peran ini dalam konfigurasi replikasi yang Anda tambahkan ke bucket *sumber* nanti. Amazon S3 mengasumsikan peran ini untuk mereplikasi objek atas nama Anda. Anda membuat peran IAM dalam dua langkah:

   - Buat peran.
   - Lampirkan kebijakan izin pada peran tersebut.

   a. Buat peran IAM.

      i. Salin kebijakan kepercayaan berikut dan simpan di berkas dengan nama `s3-role-trust-policy.json` di direktori saat ini di komputer lokal Anda. Kebijakan ini memberi Amazon S3 izin pengguna utama layanan untuk mengasumsikan peran tersebut.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
```

```
   ]
}
```

ii. Jalankan perintah berikut untuk membuat peran.

```
$ aws iam create-role \
--role-name replicationRole \
--assume-role-policy-document file://s3-role-trust-policy.json  \
--profile acctA
```

b. Lampirkan kebijakan izin pada peran tersebut.

i. Salin kebijakan kepercayaan berikut dan simpan di berkas dengan nama `s3-role-permissions-policy.json` di direktori saat ini di komputer lokal Anda. Kebijakan ini memberikan izin untuk berbagai bucket dan tindakan objek Amazon S3.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl",
            "s3:GetObjectVersionTagging"
         ],
         "Resource":[
            "arn:aws:s3:::source-bucket/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket",
            "s3:GetReplicationConfiguration"
         ],
         "Resource":[
            "arn:aws:s3:::source-bucket"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
```

```
            "s3:ReplicateObject",
            "s3:ReplicateDelete",
            "s3:ReplicateTags"
         ],
         "Resource":"arn:aws:s3:::destination-bucket/*"
      }
   ]
}
```

ii. Jalankan perintah berikut untuk membuat kebijakan dan melampirkannya ke peran.

```
$ aws iam put-role-policy \
--role-name replicationRole \
--policy-document file://s3-role-permissions-policy.json \
--policy-name replicationRolePolicy \
--profile acctA
```

5. Tambahkan konfigurasi replikasi ke bucket *source*.

   a. Meskipun Amazon S3 API memerlukan konfigurasi replikasi sebagai XHTML, Anda harus menentukan konfigurasi replikasi sebagai JSON. AWS CLI Simpan JSON berikut dalam file yang disebut `replication.json` ke direktori lokal di komputer Anda.

```
{
  "Role": "IAM-role-ARN",
  "Rules": [
    {
      "Status": "Enabled",
      "Priority": 1,
      "DeleteMarkerReplication": { "Status": "Disabled" },
      "Filter" : { "Prefix": "Tax"},
      "Destination": {
        "Bucket": "arn:aws:s3:::destination-bucket"
      }
    }
  ]
}
```

   b. Perbarui JSON dengan memberikan nilai untuk *destination-bucket* dan *IAM-role-ARN*. Simpan perubahan.

   c. Jalankan perintah berikut untuk menambahkan konfigurasi replikasi ke bucket sumber Anda. Pastikan untuk memberikan nama bucket *source*.

```
$ aws s3api put-bucket-replication \
--replication-configuration file://replication.json \
--bucket source \
--profile acctA
```

Untuk mengambil konfigurasi replikasi, gunakan perintah `get-bucket-replication`.

```
$ aws s3api get-bucket-replication \
--bucket source \
--profile acctA
```

6. Uji pengaturan di konsol Amazon S3:

   a. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

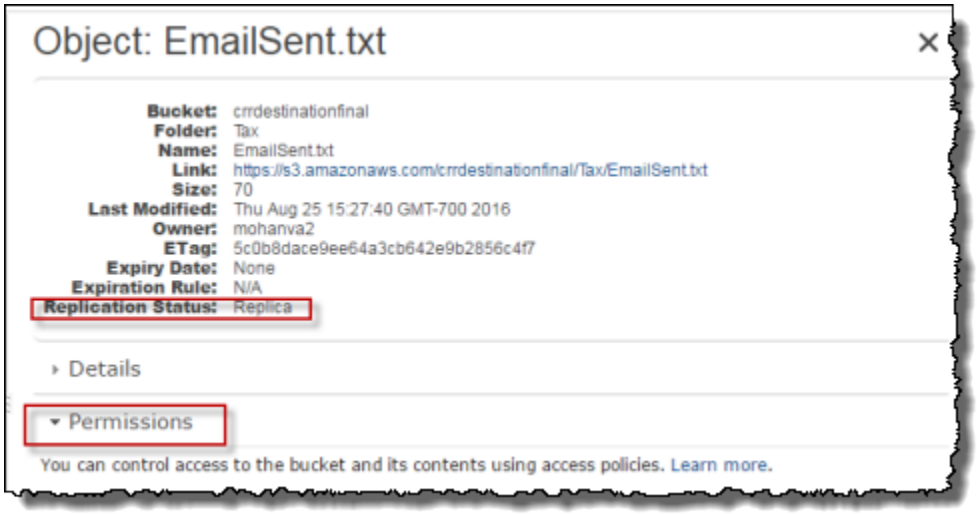
   b. Di bucket *`source`*, buat folder dengan nama `Tax`.

   c. Tambahkan sampel objek ke folder `Tax` di bucket *`source`*.

   > **Note**
   >
   > Jumlah waktu yang diperlukan Amazon S3 untuk mereplikasi objek tergantung pada ukuran objek. Untuk informasi tentang cara melihat status replikasi, lihat Mendapatkan informasi status replikasi.

   Di *`destination`* bucket, verifikasi hal berikut:

   - Bahwa Amazon S3 mereplikasi objek.
   - Dalam objek properti, bahwa Status Replikasi ditetapkan menjadi `Replica` (mengidentifikasi ini sebagai objek replika).
   - Dalam objek properti, bahwa bagian izin menunjukkan tidak ada izin. Artinya replika masih dimiliki oleh pemilik bucket *`source`*, dan pemilik bucket *`destination`* tidak memiliki izin atas replika objek. Anda dapat menambahkan konfigurasi opsional guna memberi tahu Amazon S3 untuk mengubah kepemilikan replika. Sebagai contoh, lihat Cara mengganti pemilik replika.



Menggunakan AWS SDK

Gunakan contoh kode berikut untuk menambahkan konfigurasi replikasi ke bucket dengan AWS SDK for Java dan AWS SDK for .NET, masing-masing.

Java

Contoh berikut ini menambahkan konfigurasi replikasi ke bucket dan kemudian mengambil dan memverifikasi konfigurasi. Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.identitymanagement.AmazonIdentityManagement;
import
 com.amazonaws.services.identitymanagement.AmazonIdentityManagementClientBuilder;
import com.amazonaws.services.identitymanagement.model.CreateRoleRequest;
import com.amazonaws.services.identitymanagement.model.PutRolePolicyRequest;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3Client;
import com.amazonaws.services.s3.model.BucketReplicationConfiguration;
import com.amazonaws.services.s3.model.BucketVersioningConfiguration;
import com.amazonaws.services.s3.model.CreateBucketRequest;
```

```
import com.amazonaws.services.s3.model.DeleteMarkerReplication;
import com.amazonaws.services.s3.model.DeleteMarkerReplicationStatus;
import com.amazonaws.services.s3.model.ReplicationDestinationConfig;
import com.amazonaws.services.s3.model.ReplicationRule;
import com.amazonaws.services.s3.model.ReplicationRuleStatus;
import com.amazonaws.services.s3.model.SetBucketVersioningConfigurationRequest;
import com.amazonaws.services.s3.model.StorageClass;
import com.amazonaws.services.s3.model.replication.ReplicationFilter;
import com.amazonaws.services.s3.model.replication.ReplicationFilterPredicate;
import com.amazonaws.services.s3.model.replication.ReplicationPrefixPredicate;

import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class CrossRegionReplication {

        public static void main(String[] args) throws IOException {
                Regions clientRegion = Regions.DEFAULT_REGION;
                String accountId = "*** Account ID ***";
                String roleName = "*** Role name ***";
                String sourceBucketName = "*** Source bucket name ***";
                String destBucketName = "*** Destination bucket name ***";
                String prefix = "Tax/";

                String roleARN = String.format("arn:aws:iam::%s:%s", accountId,
 roleName);
                String destinationBucketARN = "arn:aws:s3:::" + destBucketName;

                AmazonS3 s3Client = AmazonS3Client.builder()
                                .withCredentials(new ProfileCredentialsProvider())
                                .withRegion(clientRegion)
                                .build();

                createBucket(s3Client, clientRegion, sourceBucketName);
                createBucket(s3Client, clientRegion, destBucketName);
                assignRole(roleName, clientRegion, sourceBucketName,
 destBucketName);

                try {

                        // Create the replication rule.
```

```
                        List<ReplicationFilterPredicate> andOperands = new
ArrayList<ReplicationFilterPredicate>();
                        andOperands.add(new ReplicationPrefixPredicate(prefix));

                        Map<String, ReplicationRule> replicationRules = new
HashMap<String, ReplicationRule>();
                        replicationRules.put("ReplicationRule1",
                                        new ReplicationRule()
                                                        .withPriority(0)

.withStatus(ReplicationRuleStatus.Enabled)

.withDeleteMarkerReplication(
                                                                        new
DeleteMarkerReplication().withStatus(

   DeleteMarkerReplicationStatus.DISABLED))
                                                        .withFilter(new
ReplicationFilter().withPredicate(
                                                                        new
ReplicationPrefixPredicate(prefix)))
                                                        .withDestinationConfig(new
ReplicationDestinationConfig()

.withBucketARN(destinationBucketARN)

.withStorageClass(StorageClass.Standard)));

                        // Save the replication rule to the source bucket.
                        s3Client.setBucketReplicationConfiguration(sourceBucketName,
                                        new BucketReplicationConfiguration()
                                                        .withRoleARN(roleARN)

.withRules(replicationRules));

                        // Retrieve the replication configuration and verify that
the configuration
                        // matches the rule we just set.
                        BucketReplicationConfiguration replicationConfig = s3Client

.getBucketReplicationConfiguration(sourceBucketName);
                        ReplicationRule rule =
replicationConfig.getRule("ReplicationRule1");
                        System.out.println("Retrieved destination bucket ARN: "
```

```
                                        +
rule.getDestinationConfig().getBucketARN());
                        System.out.println("Retrieved priority: " +
rule.getPriority());
                        System.out.println("Retrieved source-bucket replication rule
status: " + rule.getStatus());
                } catch (AmazonServiceException e) {
                        // The call was transmitted successfully, but Amazon S3
couldn't process
                        // it, so it returned an error response.
                        e.printStackTrace();
                } catch (SdkClientException e) {
                        // Amazon S3 couldn't be contacted for a response, or the
client
                        // couldn't parse the response from Amazon S3.
                        e.printStackTrace();
                }
        }

        private static void createBucket(AmazonS3 s3Client, Regions region, String
bucketName) {
                CreateBucketRequest request = new CreateBucketRequest(bucketName,
region.getName());
                s3Client.createBucket(request);
                BucketVersioningConfiguration configuration = new
BucketVersioningConfiguration()
                                .withStatus(BucketVersioningConfiguration.ENABLED);

                SetBucketVersioningConfigurationRequest enableVersioningRequest =
new SetBucketVersioningConfigurationRequest(
                                bucketName, configuration);
                s3Client.setBucketVersioningConfiguration(enableVersioningRequest);

        }

        private static void assignRole(String roleName, Regions region, String
sourceBucket, String destinationBucket) {
                AmazonIdentityManagement iamClient =
AmazonIdentityManagementClientBuilder.standard()
                                .withRegion(region)
                                .withCredentials(new ProfileCredentialsProvider())
                                .build();
                StringBuilder trustPolicy = new StringBuilder();
                trustPolicy.append("{\\r\\n   ");
```

```
                trustPolicy.append("\\\"Version\\\":\\\"2012-10-17\\\",\\r\\n   ");
                trustPolicy.append("\\\"Statement\\\":[\\r\\n      {\\r\\n
 ");
                trustPolicy.append("\\\"Effect\\\":\\\"Allow\\\",\\r\\n         \\
\"Principal\\\":{\\r\\n            ");
                trustPolicy.append("\\\"Service\\\":\\\"s3.amazonaws.com\\\"\\r\\n
       },\\r\\n         ");
                trustPolicy.append("\\\"Action\\\":\\\"sts:AssumeRole\\\"\\r\\n
  }\\r\\n   ]\\r\\n}");

                CreateRoleRequest createRoleRequest = new CreateRoleRequest()
                                .withRoleName(roleName)

 .withAssumeRolePolicyDocument(trustPolicy.toString());

                iamClient.createRole(createRoleRequest);

                StringBuilder permissionPolicy = new StringBuilder();
                permissionPolicy.append(
                                "{\\r\\n   \\\"Version\\\":\\\"2012-10-17\\\",\\r\\n
   \\\"Statement\\\":[\\r\\n      {\\r\\n         ");
                permissionPolicy.append(
                                "\\\"Effect\\\":\\\"Allow\\\",\\r\\n         \\
\"Action\\\":[\\r\\n             ");
                permissionPolicy.append("\\\"s3:GetObjectVersionForReplication\\\",\
\r\\n            ");
                permissionPolicy.append(
                                "\\\"s3:GetObjectVersionAcl\\\"\\r\\n         ],\\r\
\n         \\\"Resource\\\":[\\r\\n            ");
                permissionPolicy.append("\\\"arn:aws:s3:::");
                permissionPolicy.append(sourceBucket);
                permissionPolicy.append("/*\\\"\\r\\n         ]\\r\\n      },\\r\\n
     {\\r\\n         ");
                permissionPolicy.append(
                                "\\\"Effect\\\":\\\"Allow\\\",\\r\\n         \\
\"Action\\\":[\\r\\n            ");
                permissionPolicy.append(
                                "\\\"s3:ListBucket\\\",\\r\\n            \\
\"s3:GetReplicationConfiguration\\\"\\r\\n         ");
                permissionPolicy.append("],\\r\\n         \\\"Resource\\\":[\\r\\n
          \\\"arn:aws:s3:::");
                permissionPolicy.append(sourceBucket);
                permissionPolicy.append("\\r\\n         ");
                permissionPolicy
```

```
                                .append("]\\r\\n      },\\r\\n      {\\r\\n
 \\\"Effect\\\":\\\"Allow\\\",\\r\\n         ");
                permissionPolicy.append(
                                "\\\"Action\\\":[\\r\\n            \\
\"s3:ReplicateObject\\\",\\r\\n            ");
                permissionPolicy
                                .append("\\\"s3:ReplicateDelete\\\",\\r\\n
  \\\"s3:ReplicateTags\\\",\\r\\n            ");
                permissionPolicy.append("\\\"s3:GetObjectVersionTagging\\\"\\r\\n\\r
\\n         ],\\r\\n         ");
                permissionPolicy.append("\\\"Resource\\\":\\\"arn:aws:s3:::");
                permissionPolicy.append(destinationBucket);
                permissionPolicy.append("/*\\\"\\r\\n      }\\r\\n   ]\\r\\n}");

                PutRolePolicyRequest putRolePolicyRequest = new
 PutRolePolicyRequest()
                                .withRoleName(roleName)
                                .withPolicyDocument(permissionPolicy.toString())
                                .withPolicyName("crrRolePolicy");

                iamClient.putRolePolicy(putRolePolicyRequest);

        }
}
```

C#

Contoh AWS SDK for .NET kode berikut menambahkan konfigurasi replikasi ke bucket dan kemudian mengambilnya. Untuk menggunakan kode ini, berikan nama untuk bucket Anda dan Amazon Resource Name (ARN) untuk peran IAM Anda. Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class CrossRegionReplicationTest
```

```
    {
        private const string sourceBucket = "*** source bucket ***";
        // Bucket ARN example - arn:aws:s3:::destinationbucket
        private const string destinationBucketArn = "*** destination bucket ARN
***";
        private const string roleArn = "*** IAM Role ARN ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint sourceBucketRegion =
RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        public static void Main()
        {
            s3Client = new AmazonS3Client(sourceBucketRegion);
            EnableReplicationAsync().Wait();
        }
        static async Task EnableReplicationAsync()
        {
            try
            {
                ReplicationConfiguration replConfig = new ReplicationConfiguration
                {
                    Role = roleArn,
                    Rules =
                        {
                            new ReplicationRule
                            {
                                Prefix = "Tax",
                                Status = ReplicationRuleStatus.Enabled,
                                Destination = new ReplicationDestination
                                {
                                    BucketArn = destinationBucketArn
                                }
                            }
                        }
                };

                PutBucketReplicationRequest putRequest = new
PutBucketReplicationRequest
                {
                    BucketName = sourceBucket,
                    Configuration = replConfig
                };
```

```
                PutBucketReplicationResponse putResponse = await
 s3Client.PutBucketReplicationAsync(putRequest);

                // Verify configuration by retrieving it.
                await RetrieveReplicationConfigurationAsync(s3Client);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
        private static async Task RetrieveReplicationConfigurationAsync(IAmazonS3
 client)
        {
            // Retrieve the configuration.
            GetBucketReplicationRequest getRequest = new GetBucketReplicationRequest
            {
                BucketName = sourceBucket
            };
            GetBucketReplicationResponse getResponse = await
 client.GetBucketReplicationAsync(getRequest);
            // Print.
            Console.WriteLine("Printing replication configuration information...");
            Console.WriteLine("Role ARN: {0}", getResponse.Configuration.Role);
            foreach (var rule in getResponse.Configuration.Rules)
            {
                Console.WriteLine("ID: {0}", rule.Id);
                Console.WriteLine("Prefix: {0}", rule.Prefix);
                Console.WriteLine("Status: {0}", rule.Status);
            }
        }
    }
}
```

Mengonfigurasi replikasi ketika bucket sumber dan tujuan dimiliki oleh akun yang berbeda

Menyiapkan replikasi ketika bucket *sumber* dan *tujuan* dimiliki oleh berbeda Akun AWS mirip dengan pengaturan replikasi ketika kedua bucket dimiliki oleh akun yang sama. Satu-satunya perbedaan adalah bahwa pemilik bucket *tujuan* harus memberikan izin kepada pemilik bucket *sumber* untuk mereplikasi objek dengan menambahkan kebijakan bucket.

Untuk informasi selengkapnya tentang mengonfigurasi replikasi menggunakan enkripsi di sisi server dengan AWS Key Management Service dalam skenario lintas akun, lihat Memberikan izin tambahan untuk skenario lintas akun.

Untuk mengonfigurasi replikasi saat bucket sumber dan tujuan dimiliki oleh yang berbeda Akun AWS

1. Dalam contoh ini, Anda membuat bucket *sumber* dan *tujuan* dalam dua yang berbeda Akun AWS. Anda harus memiliki dua profil kredensi yang ditetapkan untuk AWS CLI (dalam contoh ini, kami menggunakan `acctA` dan `acctB` untuk nama profil). Untuk informasi selengkapnya tentang mengatur profil kredensial, lihat Profil Bernama di Panduan Pengguna AWS Command Line Interface .
2. Ikuti step-by-step instruksi Mengonfigurasi bucket di akun yang sama dengan perubahan berikut:
   - Untuk semua AWS CLI perintah yang terkait dengan aktivitas bucket *sumber* (untuk membuat bucket *sumber*, mengaktifkan pembuatan versi, dan membuat peran IAM), gunakan profil. `acctA` Gunakan profil `acctB` untuk membuat bucket *tujuan*.
   - Pastikan bahwa kebijakan izin menentukan bucket *sumber* dan *tujuan* yang Anda buat untuk contoh ini.
3. Di konsol, tambahkan kebijakan bucket berikut pada bucket *tujuan* agar pemilik bucket *sumber* dapat mereplikasi objek. Pastikan untuk mengedit kebijakan dengan memberikan Akun AWS ID pemilik bucket *sumber* dan nama bucket *tujuan*.

> **Note**
>
> Untuk menggunakan contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri. Ganti *DOC-EXAMPLE-BUCKET* dengan nama bucket tujuan Anda. Ganti *source-bucket-acct-id:role/service-role/source-acct-iam-role* dengan peran yang Anda gunakan untuk konfigurasi replikasi ini.

Jika Anda membuat peran layanan IAM secara manual, tetapkan jalur peran sebagai `role/service-role/`, seperti yang ditunjukkan pada contoh kebijakan di bawah ini. Untuk informasi selengkapnya, lihat ARN IAM dalam Panduan Pengguna IAM.

```
{
   "Version":"2012-10-17",
   "Id":"",
   "Statement":[
      {
         "Sid":"Set-permissions-for-objects",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::source-bucket-acct-ID:role/service-role/source-
acct-IAM-role"
         },
         "Action":["s3:ReplicateObject", "s3:ReplicateDelete"],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      },
      {
         "Sid":"Set permissions on bucket",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::source-bucket-acct-ID:role/service-role/source-
acct-IAM-role"
         },
         "Action":["s3:GetBucketVersioning", "s3:PutBucketVersioning"],
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET"
      }
   ]
}
```

Pilih bucket dan tambahkan kebijakan bucket. Untuk petunjuk, lihat Menambahkan kebijakan bucket dengan menggunakan konsol Amazon S3.

Dalam replikasi, pemilik objek sumber adalah pemilik replika secara default. Jika bucket sumber dan tujuan dimiliki oleh yang berbeda Akun AWS, Anda dapat menambahkan pengaturan konfigurasi opsional untuk mengubah kepemilikan replika ke bucket Akun AWS yang memiliki tujuan. Ini termasuk pemberian izin `ObjectOwnerOverrideToBucketOwner`. Untuk informasi selengkapnya, lihat Mengubah pemilik replika.

Mengubah pemilik replika

Dalam replikasi, pemilik objek sumber juga memiliki replika secara default. Ketika bucket sumber dan tujuan dimiliki oleh berbeda Akun AWS dan Anda ingin mengubah kepemilikan replika ke bucket Akun AWS yang memiliki tujuan, Anda dapat menambahkan pengaturan konfigurasi opsional untuk mengubah kepemilikan replika ke bucket Akun AWS yang memiliki tujuan. Anda mungkin melakukannya, misalnya, untuk membatasi akses ke replika objek. Ini disebut sebagai opsi penggantian pemilik untuk konfigurasi replikasi. Untuk informasi selengkapnya tentang opsi penggantian pemilik, lihat Menambahkan opsi penggantian pemilik ke konfigurasi replikasi. Untuk informasi tentang mengatur konfigurasi replikasi, lihat Mereplikasi ikhtisar objek.

Untuk mengonfigurasi penggantian pemilik, Anda melakukan hal berikut:

- Tambahkan opsi penggantian pemilik ke konfigurasi replikasi guna memberi tahu Amazon S3 untuk mengubah kepemilikan replika.
- Beri Amazon S3 izin untuk mengubah kepemilikan replika.
- Tambahkan izin dalam kebijakan bucket tujuan untuk memungkinkan perubahan kepemilikan replika. Ini memungkinkan pemilik bucket tujuan menerima kepemilikan replika objek.

Untuk informasi selengkapnya, lihat Menambahkan opsi penggantian pemilik ke konfigurasi replikasi. Untuk contoh kerja dengan step-by-step instruksi, lihatCara mengganti pemilik replika.

Pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek

Saat Anda menggunakan replikasi Amazon S3 dan bucket sumber dan tujuan dimiliki oleh berbeda Akun AWS, pemilik bucket bucket tujuan dapat menonaktifkan ACL (dengan pengaturan yang diberlakukan oleh pemilik bucket untuk Kepemilikan Objek) untuk mengubah kepemilikan replika menjadi yang memiliki bucket tujuan. Akun AWS Pengaturan ini meniru perilaku penggantian pemilik yang ada tanpa perlu izin `s3:ObjectOwnerOverrideToBucketOwner`. Ini berarti bahwa semua objek yang direplikasi ke bucket tujuan dengan pengaturan yang diberlakukan pemilik bucket dimiliki oleh pemilik bucket tujuan. Untuk informasi selengkapnya tentang Kepemilikan Objek, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Menambahkan opsi penggantian pemilik ke konfigurasi replikasi

> **Warning**
>
> Tambahkan opsi penggantian pemilik hanya jika bucket sumber dan tujuan dimiliki oleh yang berbeda. Akun AWS Amazon S3 tidak memeriksa apakah bucket dimiliki oleh akun

yang sama atau berbeda. Jika Anda menambahkan penggantian pemilik saat kedua bucket dimiliki oleh yang sama Akun AWS, Amazon S3 menerapkan penggantian pemilik. Amazon S3 memberikan izin penuh kepada pemilik bucket tujuan dan tidak mereplikasi pembaruan berikutnya ke daftar kontrol akses (ACL) objek sumber. Pemilik replika dapat langsung mengubah ACL yang terkait dengan replika dengan permintaan `PUT ACL`, tetapi tidak melalui replikasi.

Untuk menentukan opsi penggantian pemilik, tambahkan berikut ini ke setiap elemen `Destination`:

- Elemen `AccessControlTranslation`, yang memberi tahu Amazon S3 untuk mengubah kepemilikan replika
- `Account`Elemen, yang menentukan Akun AWS pemilik bucket tujuan

```
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
    ...
    <Destination>
      ...
      <AccessControlTranslation>
           <Owner>Destination</Owner>
       </AccessControlTranslation>
      <Account>destination-bucket-owner-account-id</Account>
    </Destination>
  </Rule>
</ReplicationConfiguration>
```

Contoh konfigurasi replikasi berikut memberi tahu Amazon S3 untuk mereplikasi objek yang memiliki awalan kunci `Tax` ke bucket tujuan dan mengubah kepemilikan replika.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
   <Role>arn:aws:iam::account-id:role/role-name</Role>
   <Rule>
      <ID>Rule-1</ID>
      <Priority>1</Priority>
      <Status>Enabled</Status>
      <DeleteMarkerReplication>
         <Status>Disabled</Status>
      </DeleteMarkerReplication>
      <Filter>
```

```
         <Prefix>Tax</Prefix>
      </Filter>
      <Destination>
         <Bucket>arn:aws:s3:::destination-bucket</Bucket>
         <Account>destination-bucket-owner-account-id</Account>
         <AccessControlTranslation>
            <Owner>Destination</Owner>
         </AccessControlTranslation>
      </Destination>
   </Rule>
</ReplicationConfiguration>
```

Memberi Amazon S3 izin untuk mengubah kepemilikan replika

Memberi Amazon S3 izin untuk mengubah kepemilikan replika dengan menambahkan izin untuk tindakan `s3:ObjectOwnerOverrideToBucketOwner` dalam kebijakan izin yang terkait dengan peran IAM. Ini adalah peran IAM yang Anda tentukan dalam konfigurasi replikasi yang memungkinkan Amazon S3 mengasumsikan dan mereplikasi objek atas nama Anda.

```
...
{
    "Effect":"Allow",
         "Action":[
       "s3:ObjectOwnerOverrideToBucketOwner"
    ],
    "Resource":"arn:aws:s3:::destination-bucket/*"
}
...
```

Menambahkan izin dalam kebijakan bucket tujuan untuk memungkinkan perubahan kepemilikan replika

Pemilik bucket tujuan harus memberikan izin bucket sumber untuk mengubah kepemilikan replika. Pemilik bucket tujuan memberi pemilik bucket sumber izin untuk tindakan `s3:ObjectOwnerOverrideToBucketOwner`. Ini memungkinkan pemilik bucket tujuan menerima kepemilikan replika objek. Contoh pernyataan kebijakan bucket berikut menunjukkan cara melakukannya.

```
...
{
    "Sid":"1",
```

```
    "Effect":"Allow",
    "Principal":{"AWS":"source-bucket-account-id"},
    "Action":["s3:ObjectOwnerOverrideToBucketOwner"],
    "Resource":"arn:aws:s3:::destination-bucket/*"
}
...
```

Pertimbangan tambahan

Ketika Anda mengonfigurasi opsi penggantian kepemilikan, pertimbangan berikut berlaku:

- Secara default, pemilik objek sumber juga memiliki replika. Amazon S3 mereplikasi versi objek dan ACL yang terkait dengannya.

  Jika Anda menambahkan penggantian pemilik, Amazon S3 hanya mereplikasi versi objek, bukan ACL. Selain itu, Amazon S3 tidak mereplikasi perubahan berikutnya ke objek sumber ACL. Amazon S3 menetapkan ACL pada replika yang memberikan kontrol penuh kepada pemilik bucket tujuan.
- Saat Anda memperbarui konfigurasi replikasi untuk mengaktifkan atau menonaktifkan penggantian pemilik, hal berikut akan terjadi.

  - Jika Anda menambahkan opsi penggantian pemilik ke konfigurasi replikasi:

    Ketika mereplikasi versi objek, Amazon S3 membuang ACL yang terkait dengan objek sumber. Sebagai gantinya, Amazon S3 menetapkan ACL pada replika, sehingga memberikan kontrol penuh kepada pemilik bucket tujuan. Itu tidak mereplikasi perubahan berikutnya ke objek sumber ACL. Namun, perubahan ACL ini tidak berlaku untuk versi objek yang direplikasi sebelum Anda menetapkan opsi penggantian pemilik. Pembaruan ACL pada objek sumber yang direplikasi sebelum penggantian pemilik telah ditetapkan untuk terus direplikasi (karena objek dan replikanya terus memiliki pemilik yang sama).
  - Jika Anda menghapus opsi penggantian pemilik dari konfigurasi replikasi:

    Amazon S3 mereplikasi objek baru yang muncul di bucket sumber dan ACL terkait ke bucket tujuan. Untuk objek yang direplikasi sebelum Anda menghapus penggantian pemilik, Amazon S3 tidak mereplikasi ACL karena perubahan kepemilikan objek yang dibuat Amazon S3 tetap berlaku. Artinya, ACL memakai versi objek yang direplikasi ketika penggantian pemilik telah ditetapkan untuk terus tidak direplikasi.

Cara mengganti pemilik replika

Jika bucket sumber dan tujuan dalam konfigurasi replikasi dimiliki oleh berbeda Akun AWS, Anda dapat memberi tahu Amazon S3 untuk mengubah kepemilikan replika ke bucket yang memiliki tujuan. Akun AWS Contoh ini menjelaskan cara menggunakan konsol Amazon S3 dan AWS CLI untuk mengubah kepemilikan replika. Untuk informasi selengkapnya, lihat Mengubah pemilik replika.

> **Note**
>
> Saat Anda menggunakan replikasi S3 dan bucket sumber dan tujuan dimiliki oleh berbeda Akun AWS, pemilik bucket bucket tujuan dapat menonaktifkan ACL (dengan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek) untuk mengubah kepemilikan replika menjadi yang memiliki bucket tujuan. Akun AWS Pengaturan ini meniru perilaku penggantian pemilik yang ada tanpa perlu izin `s3:ObjectOwnerOverrideToBucketOwner`. Ini berarti bahwa semua objek yang direplikasi ke bucket tujuan dengan pengaturan yang diberlakukan pemilik bucket dimiliki oleh pemilik bucket tujuan. Untuk informasi selengkapnya tentang Kepemilikan Objek, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Untuk informasi selengkapnya tentang mengonfigurasi replikasi menggunakan enkripsi sisi terpisah dengan skenario lintas akun, AWS Key Management Service lihat. Memberikan izin tambahan untuk skenario lintas akun

Menggunakan konsol S3

Untuk step-by-step instruksi, lihatMengonfigurasi replikasi untuk bucket sumber dan tujuan yang dimiliki oleh akun yang sama. Topik ini memberikan instruksi untuk mengatur konfigurasi replikasi ketika bucket dimiliki oleh yang sama dan berbeda. Akun AWS

Menggunakan AWS CLI

Untuk mengubah kepemilikan replika menggunakan AWS CLI, Anda membuat bucket, mengaktifkan pembuatan versi pada bucket, membuat peran IAM yang memberikan izin Amazon S3 untuk mereplikasi objek, dan menambahkan konfigurasi replikasi ke bucket sumber. Dalam konfigurasi replikasi, Anda mengarahkan Amazon S3 untuk mengubah pemilik replika. Anda juga menguji penyiapan.

Untuk mengubah kepemilikan replika saat bucket sumber dan tujuan dimiliki oleh different Akun AWS ()AWS CLI

1. Dalam contoh ini, Anda membuat bucket *sumber* dan *tujuan* dalam dua yang berbeda Akun AWS. Konfigurasikan AWS CLI dengan dua profil bernama. Contoh ini menggunakan profil yang, masing-masing, diberi nama `acctA` dan `acctB`. Untuk informasi selengkapnya tentang mengatur profil kredensial, lihat Profil Bernama di Panduan Pengguna AWS Command Line Interface .

   > **Important**
   >
   > Profil yang Anda gunakan untuk latihan ini harus memiliki izin yang diperlukan. Misalnya, dalam konfigurasi replikasi, Anda menentukan peran IAM yang dapat diasumsikan oleh Amazon S3. Anda dapat melakukan ini hanya jika profil yang Anda gunakan memiliki izin `iam:PassRole`. Jika Anda menggunakan kredensial pengguna administrator untuk membuat profil bernama, Anda dapat melakukan semua tugas. Untuk informasi selengkapnya, lihat Memberikan Izin Pengguna untuk Meneruskan Peran ke AWS Layanan di Panduan Pengguna IAM.

   Anda harus memastikan bahwa profil ini memiliki izin yang diperlukan. Misalnya, konfigurasi replikasi mencakup peran IAM yang dapat diasumsikan oleh Amazon S3. Profil bernama yang Anda gunakan untuk melampirkan konfigurasi semacam itu ke bucket hanya dapat melakukannya jika itu memiliki izin `iam:PassRole`. Jika Anda menetapkan kredensial pengguna administrator saat membuat profil yang ditetapkan tersebut, semuanya memiliki izin. Untuk informasi selengkapnya, lihat Memberikan Izin Pengguna untuk Meneruskan Peran ke AWS Layanan di Panduan Pengguna IAM.

2. Buat bucket *sumber* dan aktifkan Penentuan Versi. Contoh ini membuat bucket *sumber* di Wilayah AS Timur (Virginia Utara) (us-east-1).

```
aws s3api create-bucket \
--bucket source \
--region us-east-1 \
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket source \
--versioning-configuration Status=Enabled \
```

```
--profile acctA
```

3. Buat bucket *tujuan* dan aktifkan Penentuan Versi. Contoh ini membuat bucket *tujuan* di Wilayah AS Barat (Oregon) (us-west-2). Gunakan profil Akun AWS yang berbeda dari yang Anda gunakan untuk bucket *sumber*.

```
aws s3api create-bucket \
--bucket destination \
--region us-west-2 \
--create-bucket-configuration LocationConstraint=us-west-2 \
--profile acctB
```

```
aws s3api put-bucket-versioning \
--bucket destination \
--versioning-configuration Status=Enabled \
--profile acctB
```

4. Anda harus menambahkan izin dalam kebijakan bucket *tujuan* untuk memungkinkan perubahan kepemilikan replika.

   a. Simpan kebijakan berikut ini ke *destination-bucket-policy*.json.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "destination_bucket_policy_sid",
      "Principal": {
        "AWS": "source-bucket-owner-account-id"
      },
      "Action": [
        "s3:ReplicateObject",
        "s3:ReplicateDelete",
        "s3:ObjectOwnerOverrideToBucketOwner",
        "s3:ReplicateTags",
        "s3:GetObjectVersionTagging"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:s3:::destination/*"
      ]
    }
```

```
  ]
}
```

b. Letakkan kebijakan di atas ke bucket *tujuan*:

```
aws s3api put-bucket-policy --region $ {destination_region} --
bucket $ {destination} --policy file://destination_bucket_policy.json
```

5. Buat peran IAM. Anda menentukan peran ini dalam konfigurasi replikasi yang Anda tambahkan ke bucket *sumber* nanti. Amazon S3 mengasumsikan peran ini untuk mereplikasi objek atas nama Anda. Anda membuat peran IAM dalam dua langkah:

   - Buat peran.
   - Lampirkan kebijakan izin pada peran tersebut.

   a. Buat peran IAM.

      i. Salin kebijakan kepercayaan berikut dan simpan di berkas dengan nama `s3-role-trust-policy.json` di direktori saat ini di komputer lokal Anda. Kebijakan ini memberi Amazon S3 izin untuk mengasumsikan peran tersebut.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

      ii. Jalankan AWS CLI perintah berikut untuk membuat peran.

```
$ aws iam create-role \
--role-name replicationRole \
--assume-role-policy-document file://s3-role-trust-policy.json  \
--profile acctA
```

b. Lampirkan kebijakan izin pada peran tersebut.

   i. Salin kebijakan kepercayaan berikut dan simpan di berkas dengan nama `s3-role-perm-pol-changeowner.json` di direktori saat ini di komputer lokal Anda. Kebijakan ini memberikan izin untuk berbagai bucket dan tindakan objek Amazon S3. Dalam langkah-langkah berikut, Anda membuat peran IAM dan melampirkan kebijakan ini pada peran tersebut.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl"
         ],
         "Resource":[
            "arn:aws:s3:::source/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket",
            "s3:GetReplicationConfiguration"
         ],
         "Resource":[
            "arn:aws:s3:::source"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ReplicateObject",
            "s3:ReplicateDelete",
            "s3:ObjectOwnerOverrideToBucketOwner",
            "s3:ReplicateTags",
            "s3:GetObjectVersionTagging"
         ],
         "Resource":"arn:aws:s3:::destination/*"
      }
```

```
    ]
}
```

ii. Untuk membuat kebijakan dan melampirkannya pada peran, jalankan perintah berikut.

```
$ aws iam put-role-policy \
--role-name replicationRole \
--policy-document file://s3-role-perm-pol-changeowner.json \
--policy-name replicationRolechangeownerPolicy \
--profile acctA
```

6. Menambahkan konfigurasi replikasi ke bucket sumber.

   a. AWS CLI Memerlukan menentukan konfigurasi replikasi sebagai JSON. Simpan JSON berikut dalam file yang disebut `replication.json` ke direktori lokal di komputer Anda. Dalam konfigurasi, penambahan `AccessControlTranslation` untuk menunjukkan perubahan dalam kepemilikan replika.

```
{
   "Role":"IAM-role-ARN",
   "Rules":[
      {
         "Status":"Enabled",
         "Priority":1,
         "DeleteMarkerReplication":{
            "Status":"Disabled"
         },
         "Filter":{
         },
         "Status":"Enabled",
         "Destination":{
            "Bucket":"arn:aws:s3:::destination",
            "Account":"destination-bucket-owner-account-id",
            "AccessControlTranslation":{
               "Owner":"Destination"
            }
         }
      }
   ]
}
```

   b. Edit JSON dengan memberikan nilai untuk ID akun pemilik bucket *`tujuan`* dan *`IAM-role-ARN`*. Simpan perubahan.

c. Untuk menambahkan konfigurasi replikasi ke bucket sumber, jalankan perintah berikut. Beri nama bucket *sumber*.

```
$ aws s3api put-bucket-replication \
--replication-configuration file://replication.json \
--bucket source \
--profile acctA
```

7. Periksa kepemilikan replika di konsol Amazon S3.

   a. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

   b. Tambahkan objek ke bucket *sumber*. *Verifikasi bahwa bucket tujuan berisi replika objek dan kepemilikan replika telah berubah menjadi bucket Akun AWS yang memiliki tujuan.*

Menggunakan AWS SDK

Untuk contoh kode guna menambahkan konfigurasi replikasi, lihat Menggunakan AWS SDK. Anda perlu memodifikasi konfigurasi replikasi dengan tepat. Untuk informasi konseptual, lihat Mengubah pemilik replika.

Memenuhi persyaratan kepatuhan menggunakan Kontrol Waktu Replikasi S3 (S3 RTC)

Kontrol Waktu Replikasi S3 (S3 RTC) membantu Anda memenuhi persyaratan kepatuhan atau bisnis untuk replikasi data dan memberikan visibilitas ke dalam waktu replikasi Amazon S3. S3 RTC mereplikasi sebagian besar objek yang Anda unggah ke Amazon S3 dalam hitungan detik, dan 99,99 persen dari objek tersebut dalam waktu 15 menit.

S3 RTC secara default mencakup metrik Replikasi S3 dan Pemberitahuan Acara Amazon S3, yang dapat Anda gunakan untuk memantau jumlah total operasi API S3 yang menunggu replikasi, ukuran total objek yang menunggu replikasi, dan waktu replikasi maksimum. Anda dapat mengaktifkan metrik replikasi secara independen dari S3 RTC. Untuk informasi selengkapnya, lihat Memantau progres dengan metrik replikasi. Selain itu, S3 RTC menyediakan peristiwa `OperationMissedThreshold` dan `OperationReplicatedAfterThreshold` yang memberi tahu pemilik bucket jika replikasi objek melebihi atau mereplikasi setelah ambang batas 15 menit.

Dengan S3 RTC, peristiwa Amazon S3 dapat memberi tahu Anda secara instans ketika objek tidak mereplikasi dalam 15 menit dan saat objek mereplikasi setelah ambang batas 15 menit. Acara

Amazon S3 tersedia melalui Amazon SQS, Amazon SNS, atau. AWS Lambda Untuk informasi selengkapnya, lihat the section called “Notifikasi Peristiwa Amazon S3”.

Topik

- Kontrol Waktu Replikasi S3
- Metrik replikasi dengan RTC S3
- Menggunakan notifikasi peristiwa Amazon S3 untuk melacak objek replikasi
- Praktik terbaik dan pedoman untuk S3 RTC
- Mengaktifkan Kontrol Waktu Replikasi S3 (S3 RTC)

Kontrol Waktu Replikasi S3

Anda dapat mulai menggunakan Kontrol Waktu Replikasi S3 (S3 RTC) dengan aturan replikasi baru atau yang sudah ada. Anda dapat memilih untuk menerapkan aturan replikasi Anda ke seluruh bucket S3, atau ke objek Amazon S3 dengan awalan atau tag tertentu. Saat Anda mengaktifkan S3 RTC, metrik replikasi juga diaktifkan pada aturan replikasi Anda.

Jika Anda menggunakan versi terbaru konfigurasi replikasi (yaitu, Anda menentukan elemen `Filter` dalam aturan konfigurasi replikasi), Amazon S3 tidak mereplikasi penanda hapus secara default. Namun Anda dapat menambahkan replikasi penanda hapus ke non-tag-based aturan.

> **Note**
>
> Metrik replikasi ditagih dengan tarif yang sama dengan metrik kustom Amazon CloudWatch . Untuk informasi, lihat harga Amazon CloudWatch.

Untuk informasi selengkapnya tentang membuat aturan dengan S3 RTC, lihat Mengaktifkan Kontrol Waktu Replikasi S3 (S3 RTC).

Metrik replikasi dengan RTC S3

Aturan replikasi dengan Kontrol Waktu Replikasi S3 (S3 RTC) memungkinkan publikasi metrik replikasi. Dengan metrik replikasi, Anda dapat memantau jumlah total operasi API S3 yang masih menunggu replikasi, ukuran total objek yang menunggu replikasi, waktu replikasi maksimum ke Wilayah tujuan, dan jumlah total operasi yang gagal mereplikasi. Kemudian, Anda dapat memantau setiap set data yang akan direplikasi secara terpisah.

Metrik replikasi tersedia dalam 15 menit setelah S3 RTC diaktifkan. Metrik replikasi tersedia melalui konsol Amazon S3, API Amazon S3, SDK, (), AWS dan Amazon.AWS Command Line InterfaceAWS CLI CloudWatch Untuk informasi selengkapnya, lihat Memantau metrik dengan Amazon CloudWatch.

Untuk informasi selengkapnya tentang menemukan metrik replikasi di konsol S3, lihat Melihat metrik replikasi dengan menggunakan konsol Amazon S3.

Menggunakan notifikasi peristiwa Amazon S3 untuk melacak objek replikasi

Anda dapat melacak waktu replikasi untuk objek yang tidak mereplikasi dalam 15 menit dengan memantau pemberitahuan peristiwa khusus yang dipublikasikan oleh Kontrol Waktu Replikasi S3 (S3 RTC). Peristiwa ini diterbitkan ketika objek yang memenuhi syarat untuk replikasi menggunakan S3 RTC tidak mereplikasi dalam 15 menit, dan saat objek mereplikasi setelah ambang batas 15 menit.

Metrik replikasi tersedia dalam 15 menit setelah S3 RTC diaktifkan. Acara Amazon S3 tersedia melalui Amazon SQS, Amazon SNS, atau. AWS Lambda Untuk informasi selengkapnya, lihat Notifikasi Peristiwa Amazon S3.

Praktik terbaik dan pedoman untuk S3 RTC

Ketika mereplikasi data di Amazon S3 menggunakan Kontrol Waktu Replikasi S3 (S3 RTC), ikuti pedoman praktik terbaik ini untuk mengoptimalkan kinerja replikasi untuk beban kerja Anda.

Topik

- Replikasi Amazon S3 dan pedoman kinerja tingkat permintaan
- Memperkirakan tingkat permintaan replikasi Anda
- Melampaui batas kecepatan transfer data S3 RTC
- AWS KMS tingkat permintaan replikasi objek terenkripsi

Replikasi Amazon S3 dan pedoman kinerja tingkat permintaan

Ketika mengunggah dan mengambil penyimpanan dari Amazon S3, aplikasi Anda dapat mencapai ribuan transaksi per detik dalam kinerja permintaan. Misalnya, aplikasi dapat mencapai setidaknya 3.500 permintaan PUT/COPY/POST/DELETE atau 5.500 permintaan GET/HEAD per detik per awalan dalam bucket S3, termasuk permintaan yang dibuat replikasi S3 atas nama Anda. Tidak ada batas jumlah prefiks dalam bucket. Anda dapat meningkatkan kinerja baca atau tulis dengan cara menyelaraskan pembacaan. Misalnya, jika Anda membuat 10 awalan dalam bucket S3 untuk menyelaraskan pembacaan, Anda dapat menskalakan kinerja baca Anda menjadi 55.000 permintaan baca per detik.

Amazon S3 secara otomatis menskalakan ke dalam tingkat permintaan berkelanjutan di atas panduan ini, atau tingkat permintaan berkelanjutan yang bersamaan dengan permintaan LIST. Meski Amazon S3 secara internal mengoptimalkan tingkat permintaan baru, Anda mungkin menerima respons permintaan HTTP 503 untuk sementara hingga optimasi selesai. Ini mungkin terjadi dengan kenaikan permintaan per detik, atau ketika Anda pertama kali mengaktifkan S3 RTC. Selama periode ini, latensi replikasi Anda dapat meningkat. Perjanjian tingkat layanan (SLA) S3 RTC tidak berlaku untuk periode waktu ketika pedoman kinerja Amazon S3 pada permintaan per detik terlampaui.

SLA S3 RTC juga tidak berlaku selama periode waktu ketika kecepatan transfer data replikasi Anda melebihi batas default 1 Gbps. Jika Anda memperkirakan kecepatan transfer replikasi Anda melebihi 1 Gbps, Anda dapat menghubungi AWS Support Pusat atau menggunakan Service Quotas untuk meminta peningkatan batas Anda.

Memperkirakan tingkat permintaan replikasi Anda

Tingkat permintaan total Anda termasuk permintaan yang dibuat oleh replikasi Amazon S3 atas nama Anda harus berada dalam pedoman tarif permintaan Amazon S3 untuk bucket sumber dan tujuan replikasi. Untuk setiap objek yang direplikasi, replikasi Amazon S3 membuat hingga lima permintaan GET/HEAD dan satu permintaan PUT ke bucket sumber, dan satu permintaan PUT ke setiap bucket tujuan.

Misalnya, jika Anda berharap untuk mereplikasi 100 objek per detik, replikasi Amazon S3 mungkin melakukan 100 permintaan PUT tambahan atas nama Anda dengan total 200 PUT per detik ke bucket S3 sumber. Replikasi Amazon S3 juga dapat melakukan hingga 500 GET/HEAD (5 permintaan GET/HEAD untuk setiap objek yang direplikasi.)

> **Note**
>
> Anda hanya menanggung biaya untuk satu permintaan PUT per objek yang direplikasi. Untuk informasi selengkapnya, lihat informasi harga di FAQ Amazon S3 tentang replikasi.

Melampaui batas kecepatan transfer data S3 RTC

Jika Anda memperkirakan kecepatan transfer data Kontrol Waktu Replikasi S3 Anda melebihi batas default 1 Gbps, hubungi AWS Support Pusat atau gunakan Service Quotas untuk meminta peningkatan batas Anda.

AWS KMS tingkat permintaan replikasi objek terenkripsi

Saat Anda mereplikasi objek yang dienkripsi dengan enkripsi sisi server (SSE-KMS) menggunakan replikasi Amazon S3, () batas permintaan per detik berlaku. AWS Key Management Service AWS KMS AWS KMS mungkin menolak permintaan yang valid karena tingkat permintaan Anda melebihi batas jumlah permintaan per detik. Ketika permintaan dibatasi, AWS KMS mengembalikan kesalahan. `ThrottlingException` AWS KMS Batas tarif permintaan berlaku untuk permintaan yang Anda buat secara langsung dan permintaan yang dibuat oleh replikasi Amazon S3 atas nama Anda.

Misalnya, jika Anda berharap untuk mereplikasi 1.000 objek per detik, Anda dapat mengurangi 2.000 permintaan dari batas tingkat AWS KMS permintaan Anda. Tingkat permintaan per detik yang dihasilkan tersedia untuk AWS KMS beban kerja Anda tidak termasuk replikasi. Anda dapat menggunakan metrik AWS KMS permintaan di Amazon CloudWatch untuk memantau total tingkat AWS KMS permintaan pada Anda Akun AWS.

Mengaktifkan Kontrol Waktu Replikasi S3 (S3 RTC)

Kontrol Waktu Replikasi S3 (S3 RTC) membantu Anda memenuhi persyaratan kepatuhan atau bisnis untuk replikasi data dan memberikan visibilitas ke dalam waktu replikasi Amazon S3. S3 RTC mereplikasi sebagian besar objek yang Anda unggah ke Amazon S3 dalam hitungan detik, dan 99,99 persen dari objek tersebut dalam waktu 15 menit.

Dengan S3 RTC, Anda dapat memantau jumlah dan ukuran total objek yang masih menunggu replikasi, dan waktu replikasi maksimal ke Wilayah tujuan. Metrik replikasi tersedia melalui CloudWatch Panduan Pengguna AWS Management ConsoleAmazon dan Amazon. Untuk informasi selengkapnya, lihat the section called “Metrik replikasi S3 di CloudWatch ”.

Menggunakan konsol S3

Untuk step-by-step instruksi, lihatMengonfigurasi replikasi untuk bucket sumber dan tujuan yang dimiliki oleh akun yang sama. Topik ini memberikan instruksi untuk mengaktifkan S3 RTC dalam konfigurasi replikasi Anda saat bucket dimiliki oleh yang sama dan berbeda. Akun AWS

Menggunakan AWS CLI

Untuk menggunakan AWS CLI untuk mereplikasi objek dengan S3 RTC diaktifkan, Anda membuat bucket, mengaktifkan pembuatan versi pada bucket, membuat peran IAM yang memberikan izin Amazon S3 untuk mereplikasi objek, dan menambahkan konfigurasi replikasi ke bucket sumber. Konfigurasi replikasi perlu mengaktifkan Kontrol Waktu Replikasi S3 (S3 RTC).

Untuk mereplikasi dengan S3 RTC diaktifkan (AWS CLI)

- Contoh berikut menetapkan `ReplicationTime` dan `Metric`, dan menambahkan konfigurasi replikasi ke bucket sumber.

```
{
    "Rules": [
        {
            "Status": "Enabled",
            "Filter": {
                "Prefix": "Tax"
            },
            "DeleteMarkerReplication": {
                "Status": "Disabled"
            },
            "Destination": {
                "Bucket": "arn:aws:s3:::destination",
                "Metrics": {
                    "Status": "Enabled",
                    "EventThreshold": {
                        "Minutes": 15
                    }
                },
                "ReplicationTime": {
                    "Status": "Enabled",
                    "Time": {
                        "Minutes": 15
                    }
                }
            },
            "Priority": 1
        }
    ],
    "Role": "IAM-Role-ARN"
}
```

> **Important**
>
> `Metrics:EventThreshold:Minutes` dan `ReplicationTime:Time:Minutes` hanya dapat memiliki 15 sebagai nilai valid.

Menggunakan AWS SDK for Java

Contoh Java berikut menambahkan konfigurasi replikasi dengan Kontrol Waktu Replikasi S3 (S3 RTC).

```
import software.amazon.awssdk.auth.credentials.AwsBasicCredentials;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.model.DeleteMarkerReplication;
import software.amazon.awssdk.services.s3.model.Destination;
import software.amazon.awssdk.services.s3.model.Metrics;
import software.amazon.awssdk.services.s3.model.MetricsStatus;
import software.amazon.awssdk.services.s3.model.PutBucketReplicationRequest;
import software.amazon.awssdk.services.s3.model.ReplicationConfiguration;
import software.amazon.awssdk.services.s3.model.ReplicationRule;
import software.amazon.awssdk.services.s3.model.ReplicationRuleFilter;
import software.amazon.awssdk.services.s3.model.ReplicationTime;
import software.amazon.awssdk.services.s3.model.ReplicationTimeStatus;
import software.amazon.awssdk.services.s3.model.ReplicationTimeValue;

public class Main {

  public static void main(String[] args) {
    S3Client s3 = S3Client.builder()
      .region(Region.US_EAST_1)
      .credentialsProvider(() -> AwsBasicCredentials.create(
          "AWS_ACCESS_KEY_ID",
          "AWS_SECRET_ACCESS_KEY")
      )
      .build();

    ReplicationConfiguration replicationConfig = ReplicationConfiguration
      .builder()
      .rules(
          ReplicationRule
            .builder()
            .status("Enabled")
            .priority(1)
            .deleteMarkerReplication(
                DeleteMarkerReplication
                    .builder()
                    .status("Disabled")
                    .build()
            )
            .destination(
```

```
                Destination
                    .builder()
                    .bucket("destination_bucket_arn")
                    .replicationTime(
                        ReplicationTime.builder().time(
                            ReplicationTimeValue.builder().minutes(15).build()
                        ).status(
                            ReplicationTimeStatus.ENABLED
                        ).build()
                    )
                    .metrics(
                        Metrics.builder().eventThreshold(
                            ReplicationTimeValue.builder().minutes(15).build()
                        ).status(
                            MetricsStatus.ENABLED
                        ).build()
                    )
                    .build()
            )
            .filter(
                ReplicationRuleFilter
                    .builder()
                    .prefix("testtest")
                    .build()
            )
        .build())
        .role("role_arn")
        .build();

    // Put replication configuration
    PutBucketReplicationRequest putBucketReplicationRequest =
 PutBucketReplicationRequest
      .builder()
      .bucket("source_bucket")
      .replicationConfiguration(replicationConfig)
      .build();

    s3.putBucketReplication(putBucketReplicationRequest);
  }
}
```

Untuk informasi selengkapnya, lihat Memenuhi persyaratan kepatuhan menggunakan Kontrol Waktu Replikasi S3 (S3 RTC).

Mereplikasi objek terenkripsi (SSE-S3, SSE-KMS, SSE-KMS)

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Ada beberapa pertimbangan khusus ketika Anda mereplikasi objek yang telah dienkripsi dengan menggunakan enkripsi di sisi server. Amazon S3 mendukung tipe enkripsi di sisi server berikut:

- Enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3)
- Enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS)
- Enkripsi sisi server dua lapis dengan kunci (DSSE-KMS) AWS KMS
- Enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C)

Untuk informasi lebih lanjut tentang enkripsi di sisi server, lihat the section called “enkripsi di sisi server”.

Topik ini menjelaskan izin yang Anda perlukan untuk mengarahkan Amazon S3 guna mereplikasi objek yang telah dienkripsi menggunakan enkripsi di sisi server. Topik ini juga menyediakan elemen konfigurasi tambahan yang dapat Anda tambahkan dan contoh kebijakan AWS Identity and Access Management (IAM) yang memberikan izin yang diperlukan untuk mereplikasi objek terenkripsi.

Untuk contoh dengan step-by-step instruksi, lihatMengaktifkan replikasi untuk objek terenkripsi. Untuk informasi tentang pembuatan konfigurasi replikasi, lihat Mereplikasi ikhtisar objek.

> **Note**
>
> Anda dapat menggunakan Multi-wilayah AWS KMS keys di Amazon S3. Namun, Amazon S3 saat ini memperlakukan kunci multi-Wilayah selayaknya kunci satu Wilayah, dan tidak menggunakan fitur multi-Wilayah dari kunci tersebut. Untuk informasi selengkapnya, lihat Menggunakan kunci multi-Wilayah di Panduan Pengembang AWS Key Management Service .

Topik

- Bagaimana enkripsi bucket default memengaruhi replikasi
- Mereplikasi objek yang dienkripsi dengan SSE-C
- Mereplikasi objek yang dienkripsi dengan SSE-S3, SSE-KMS, atau DSSE-KMS
- Mengaktifkan replikasi untuk objek terenkripsi

Bagaimana enkripsi bucket default memengaruhi replikasi

Saat Anda mengaktifkan enkripsi default untuk bucket tujuan replikasi, perilaku enkripsi berikut berlaku:

- Jika objek dalam bucket sumber tidak dienkripsi, objek replika dalam bucket tujuan akan dienkripsi menggunakan pengaturan enkripsi default dari bucket tujuan. Akibatnya, tag entitas (ETag) dari objek sumber berbeda dari ETag objek replika. Jika Anda memiliki aplikasi yang menggunakan ETag, Anda harus memperbarui aplikasi tersebut untuk memperhitungkan perbedaan ini.
- Jika objek dalam bucket sumber dienkripsi dengan menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3), enkripsi sisi server dengan kunci () (SSE-KMS), atau enkripsi sisi server dua lapis dengan AWS Key Management Service kunci (DSSE-KMS), objek replika di bucket tujuan menggunakan jenis enkripsi yang sama dengan AWS KMS objek sumber.AWS KMS Pengaturan enkripsi default bucket tujuan tidak digunakan.

Mereplikasi objek yang dienkripsi dengan SSE-C

Dengan menggunakan enkripsi sisi server dengan kunci yang disediakan pelanggan (SSE-C), Anda dapat mengelola kunci enkripsi milik Anda sendiri. Dengan SSE-C, Anda mengelola kunci sedangkan Amazon S3 mengelola proses enkripsi dan dekripsi. Anda harus memberikan kunci enkripsi sebagai bagian dari permintaan Anda, tetapi Anda tidak perlu menulis kode apa pun untuk

melakukan enkripsi atau dekripsi objek. Saat mengunggah sebuah objek, Amazon S3 mengenkripsi objek menggunakan kunci yang Anda berikan. Amazon S3 kemudian membersihkan kunci itu dari memori. Saat mengambil sebuah objek, Anda harus memberikan kunci enkripsi yang sama sebagai bagian dari permintaan Anda. Untuk informasi selengkapnya, lihat the section called “Kunci enkripsi yang disediakan pelanggan (SSE-C)”.

S3 Replikasi mendukung objek yang dienkripsi dengan SSE-C. Anda dapat mengonfigurasi replikasi objek SSE-C di konsol Amazon S3 atau dengan AWS SDK, dengan cara yang sama seperti Anda mengonfigurasi replikasi untuk objek yang tidak terenkripsi. Tidak ada izin SSE-C tambahan di luar apa yang saat ini diperlukan untuk replikasi.

Replikasi S3 secara otomatis mereplikasi objek terenkripsi SSE-C yang baru diunggah jika memenuhi syarat, seperti yang ditentukan dalam konfigurasi Replikasi S3 Anda. Untuk mereplikasi objek yang ada di bucket Anda, gunakan Replikasi Batch S3. Untuk informasi selengkapnya tentang mereplikasi objek, lihat the section called “Menyiapkan replikasi langsung” dan the section called “Mereplikasi objek yang ada”.

Tidak ada biaya tambahan untuk mereplikasi objek SSE-C. Untuk detail tentang harga replikasi, lihat halaman harga Amazon S3.

Mereplikasi objek yang dienkripsi dengan SSE-S3, SSE-KMS, atau DSSE-KMS

Secara default, Amazon S3 tidak mereplikasi objek yang dienkripsi dengan SSE-KMS atau DSSE-KMS. Bagian ini menjelaskan elemen konfigurasi tambahan yang dapat Anda tambahkan ke Amazon S3 langsung untuk mereplikasi objek tersebut.

Untuk contoh dengan step-by-step instruksi, lihatMengaktifkan replikasi untuk objek terenkripsi. Untuk informasi tentang pembuatan konfigurasi replikasi, lihat Mereplikasi ikhtisar objek.

Menentukan informasi tambahan dalam konfigurasi replikasi

Dalam konfigurasi replikasi, Anda melakukan hal berikut:

- Dalam `Destination` elemen dalam konfigurasi replikasi Anda, tambahkan ID kunci terkelola AWS KMS pelanggan simetris yang Anda ingin Amazon S3 gunakan untuk mengenkripsi replika objek, seperti yang ditunjukkan dalam contoh konfigurasi replikasi berikut.
- Pilih ikut secara eksplisit dengan mengaktifkan replikasi objek yang dienkripsi menggunakan kunci KMS (SSE-KMS atau DSSE-KMS). Untuk memilih ikut, tambahkan elemen `SourceSelectionCriteria`, seperti yang ditunjukkan dalam konfigurasi replikasi contoh berikut.

```
<ReplicationConfiguration>
   <Rule>
      ...
      <SourceSelectionCriteria>
         <SseKmsEncryptedObjects>
           <Status>Enabled</Status>
         </SseKmsEncryptedObjects>
      </SourceSelectionCriteria>

      <Destination>
          ...
          <EncryptionConfiguration>
             <ReplicaKmsKeyID>AWS KMS key ARN or Key Alias ARN that's in the same
 Wilayah AWS as the destination bucket.</ReplicaKmsKeyID>
          </EncryptionConfiguration>
       </Destination>
      ...
   </Rule>
</ReplicationConfiguration>
```

> **Important**
>
> Kunci KMS harus dibuat Wilayah AWS sama dengan ember tujuan.
> Kunci KMS harus valid. Operasi API `PutBucketReplication` tidak memeriksa validitas kunci KMS. Jika menggunakan kunci KMS yang tidak valid, Anda akan menerima kode status `200 OK` HTTP sebagai respons, tetapi replikasi gagal.

Contoh berikut menunjukkan konfigurasi replikasi yang meliputi elemen konfigurasi opsional. Konfigurasi replikasi ini memiliki satu aturan. Aturan ini berlaku untuk objek dengan awalan kunci `Tax`. Amazon S3 menggunakan ID AWS KMS key tertentu untuk mengenkripsi replika objek ini.

```
<?xml version="1.0" encoding="UTF-8"?>
<ReplicationConfiguration>
   <Role>arn:aws:iam::account-id:role/role-name</Role>
   <Rule>
      <ID>Rule-1</ID>
      <Priority>1</Priority>
      <Status>Enabled</Status>
      <DeleteMarkerReplication>
```

```
         <Status>Disabled</Status>
      </DeleteMarkerReplication>
      <Filter>
         <Prefix>Tax</Prefix>
      </Filter>
      <Destination>
         <Bucket>arn:aws:s3:::example-s3-destination-bucket</Bucket>
         <EncryptionConfiguration>
            <ReplicaKmsKeyID>AWS KMS key ARN or Key Alias ARN that's in the
 same Wilayah AWS as the destination bucket. (S3 uses this key to encrypt object
 replicas.)</ReplicaKmsKeyID>
         </EncryptionConfiguration>
      </Destination>
      <SourceSelectionCriteria>
         <SseKmsEncryptedObjects>
            <Status>Enabled</Status>
         </SseKmsEncryptedObjects>
      </SourceSelectionCriteria>
   </Rule>
</ReplicationConfiguration>
```

Memberikan izin tambahan untuk peran IAM

Untuk mereplikasi objek yang dienkripsi saat istirahat menggunakan SSE-S3, SSE-KMS, atau DSSE-KMS, berikan izin tambahan berikut ke peran (IAM) yang Anda tentukan dalam konfigurasi replikasi. AWS Identity and Access Management Anda memberikan izin ini dengan memperbarui kebijakan izin yang terkait dengan peran IAM.

- Tindakan **`s3:GetObjectVersionForReplication`** untuk objek sumber–Tindakan ini memungkinkan Amazon S3 mereplikasi objek tidak terenkripsi dan objek yang dibuat dengan enkripsi di sisi server menggunakan SSE-S3, SSE-KMS, atau DSSE-KMS.

> **Note**
>
> Kami menyarankan Anda menggunakan tindakan `s3:GetObjectVersionForReplication` alih-alih tindakan `s3:GetObjectVersion` karena `s3:GetObjectVersionForReplication` hanya memberi Amazon S3 izin minimum yang diperlukan untuk replikasi. Selain itu, tindakan `s3:GetObjectVersion` memungkinkan replikasi objek tidak terenkripsi dan terenkripsi SSE-S3, tetapi bukan objek yang dienkripsi menggunakan kunci KMS (SSE-KMS atau DSSE-KMS).

- **`kms:Decrypt`**dan **`kms:Encrypt`** AWS KMS tindakan untuk kunci KMS
  - Anda harus memberikan izin `kms:Decrypt` untuk AWS KMS key yang digunakan untuk mendekripsi objek sumber.
  - Anda harus memberikan izin `kms:Encrypt` untuk AWS KMS key yang digunakan untuk mengenkripsi replika objek.
- Tindakan **`kms:GenerateDataKey`** untuk mereplikasi objek plaintext–Jika Anda mereplikasi objek plaintext ke bucket dengan enkripsi SSE-KMS atau DSSE-KMS yang diaktifkan secara default, Anda harus menyertakan izin `kms:GenerateDataKey` untuk konteks enkripsi tujuan dan kunci KMS dalam kebijakan IAM.

Kami menyarankan Anda membatasi izin ini hanya untuk bucket tujuan dan objek dengan menggunakan AWS KMS tombol kondisi. Akun AWS Yang memiliki peran IAM harus memiliki izin untuk `kms:Encrypt` dan `kms:Decrypt` tindakan untuk kunci KMS yang tercantum dalam kebijakan. Jika kunci KMS dimiliki oleh orang lain Akun AWS, pemilik kunci KMS harus memberikan izin ini kepada Akun AWS yang memiliki peran IAM. Untuk informasi selengkapnya tentang mengelola akses ke kunci KMS ini, lihat Menggunakan Kebijakan IAM dengan Panduan AWS KMSAWS Key Management Service Pengembang.

Replikasi dan Kunci Bucket S3

Untuk menggunakan replikasi dengan S3 Bucket Key, AWS KMS key kebijakan untuk kunci KMS yang digunakan untuk mengenkripsi replika objek harus menyertakan `kms:Decrypt` izin untuk prinsipal panggilan. Panggilan ke `kms:Decrypt` memverifikasi integritas Kunci Bucket S3 sebelum menggunakannya. Untuk informasi selengkapnya, lihat Menggunakan kunci Bucket S3 dengan replikasi.

Saat Kunci Bucket S3 diaktifkan untuk bucket sumber atau tujuan, konteks enkripsi akan menjadi Amazon Resource Name (ARN) bucket, bukan ARN objek (misalnya, `arn:aws:s3:::`*`bucket_ARN`*). Anda harus memperbarui kebijakan IAM Anda untuk menggunakan ARN bucket untuk konteks enkripsi:

```
"kms:EncryptionContext:aws:s3:arn": [
"arn:aws:s3:::bucket_ARN"
]
```

Untuk informasi selengkapnya, lihat Konteks enkripsi (`x-amz-server-side-encryption-context`) (di bagian “Menggunakan API REST”) dan Perubahan yang perlu diperhatikan sebelum mengaktifkan Kunci Bucket S3.

Contoh kebijakan–Menggunakan SSE-S3 dan SSE-KMS dengan replikasi

Contoh kebijakan IAM berikut menunjukkan pernyataan untuk menggunakan SSE-S3 dan SSE-KMS dengan replikasi.

Example – Menggunakan SSE-KMS dengan bucket tujuan terpisah

Contoh kebijakan berikut menunjukkan pernyataan untuk menggunakan SSE-KMS dengan bucket tujuan terpisah.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Action": ["kms:Decrypt"],
         "Effect": "Allow",
         "Condition": {
             "StringLike": {
                  "kms:ViaService": "s3.source-bucket-region.amazonaws.com",
                  "kms:EncryptionContext:aws:s3:arn": [
                      "arn:aws:s3:::example-s3-source-bucket/key-prefix1*"
                  ]
             }
          },
          "Resource": [
             "List of AWS KMS key ARNs that are used to encrypt source objects."
          ]
      },
      {
         "Action": ["kms:Encrypt"],
         "Effect": "Allow",
         "Condition": {
            "StringLike": {
               "kms:ViaService": "s3.destination-bucket-1-region.amazonaws.com",
               "kms:EncryptionContext:aws:s3:arn": [
                  "arn:aws:s3:::example-s3-destination-bucket1/key-prefix1*"
               ]
         }
         },
```

```
            "Resource": [
               "AWS KMS key ARNs (in the same Wilayah AWS as destination bucket 1). Used
 to encrypt object replicas created in destination bucket 1."
            ]
         },
         {
            "Action": ["kms:Encrypt"],
            "Effect": "Allow",
            "Condition": {
               "StringLike": {
                  "kms:ViaService": "s3.destination-bucket-2-region.amazonaws.com",
                  "kms:EncryptionContext:aws:s3:arn": [
                     "arn:aws:s3:::example-s3-destination-bucket2/key-prefix1*"
                  ]
            }
            },
            "Resource": [
               "AWS KMS key ARNs (in the same Wilayah AWS as destination bucket 2). Used
 to encrypt object replicas created in destination bucket 2."
            ]
         }
      ]
}
```

Example – Mereplikasi objek yang dibuat dengan SSE-S3 dan SSE-KMS

Berikut ini adalah kebijakan IAM lengkap yang memberikan izin yang diperlukan untuk mereplikasi objek tidak terenkripsi, objek yang dibuat dengan SSE-S3, dan objek yang dibuat dengan SSE-KMS.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetReplicationConfiguration",
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::example-s3-source-bucket"
         ]
      },
      {
```

```
            "Effect":"Allow",
            "Action":[
               "s3:GetObjectVersionForReplication",
               "s3:GetObjectVersionAcl"
            ],
            "Resource":[
               "arn:aws:s3:::example-s3-source-bucket/key-prefix1*"
            ]
         },
         {
            "Effect":"Allow",
            "Action":[
               "s3:ReplicateObject",
               "s3:ReplicateDelete"
            ],
            "Resource":"arn:aws:s3:::example-s3-destination-bucket/key-prefix1*"
         },
         {
            "Action":[
               "kms:Decrypt"
            ],
            "Effect":"Allow",
            "Condition":{
               "StringLike":{
                  "kms:ViaService":"s3.source-bucket-region.amazonaws.com",
                  "kms:EncryptionContext:aws:s3:arn":[
                     "arn:aws:s3:::example-s3-source-bucket/key-prefix1*"
                  ]
               }
            },
            "Resource":[
              "List of the AWS KMS key ARNs that are used to encrypt source objects."
            ]
         },
         {
            "Action":[
               "kms:Encrypt"
            ],
            "Effect":"Allow",
            "Condition":{
               "StringLike":{
                  "kms:ViaService":"s3.destination-bucket-region.amazonaws.com",
                  "kms:EncryptionContext:aws:s3:arn":[
                     "arn:aws:s3:::example-s3-destination-bucket/prefix1*"
```

```
                ]
            }
        },
        "Resource":[
            "AWS KMS key ARNs (in the same Wilayah AWS as the destination bucket) to
 use for encrypting object replicas"
        ]
    }
  ]
}
```

Example – Mereplikasi objek dengan Kunci Bucket S3

Berikut ini adalah kebijakan IAM lengkap yang memberikan izin yang diperlukan untuk mereplikasi objek dengan Kunci Bucket S3.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetReplicationConfiguration",
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::example-s3-source-bucket"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl"
         ],
         "Resource":[
            "arn:aws:s3:::example-s3-source-bucket/key-prefix1*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:ReplicateObject",
```

```
            "s3:ReplicateDelete"
         ],
         "Resource":"arn:aws:s3:::example-s3-destination-bucket/key-prefix1*"
      },
      {
         "Action":[
            "kms:Decrypt"
         ],
         "Effect":"Allow",
         "Condition":{
            "StringLike":{
               "kms:ViaService":"s3.source-bucket-region.amazonaws.com",
               "kms:EncryptionContext:aws:s3:arn":[
                  "arn:aws:s3:::example-s3-source-bucket"
               ]
            }
         },
         "Resource":[
           "List of the AWS KMS key ARNs that are used to encrypt source objects."
         ]
      },
      {
         "Action":[
            "kms:Encrypt"
         ],
         "Effect":"Allow",
         "Condition":{
            "StringLike":{
               "kms:ViaService":"s3.destination-bucket-region.amazonaws.com",
               "kms:EncryptionContext:aws:s3:arn":[
                  "arn:aws:s3:::example-s3-destination-bucket"
               ]
            }
         },
         "Resource":[
            "AWS KMS key ARNs (in the same Wilayah AWS as the destination bucket) to
 use for encrypting object replicas"
         ]
      }
   ]
}
```

Memberikan izin tambahan untuk skenario lintas akun

Dalam skenario lintas akun, di mana bucket sumber dan tujuan dimiliki oleh yang berbeda Akun AWS, Anda dapat menggunakan kunci KMS untuk mengenkripsi replika objek. Namun, pemilik kunci KMS harus memberikan izin pemilik bucket sumber untuk menggunakan kunci KMS.

> **Note**
>
> Jika Anda perlu mereplikasi data lintas akun SSE-KMS, maka aturan replikasi Anda harus menentukan kunci yang dikelola pelanggan untuk akun tujuan. AWS KMS Kunci yang dikelola AWSjangan izinkan penggunaan lintas akun, dan karenanya tidak dapat digunakan untuk melakukan replikasi lintas akun.

Untuk memberikan izin kepada pemilik bucket sumber untuk menggunakan kunci KMS (konsol AWS KMS)

1. Masuk ke AWS Management Console dan buka AWS KMS konsol di https://console.aws.amazon.com/kms.
2. Untuk mengubah Wilayah AWS, gunakan pemilih Wilayah di sudut kanan atas halaman.
3. Untuk melihat tombol di akun yang Anda buat dan kelola, di panel navigasi pilih CMK.
4. Pilih tombol KMS.
5. Di bawah bagian Konfigurasi umum, pilih tab Kebijakan kunci.
6. Gulir ke bawah ke Lainnya Akun AWS.
7. Pilih Tambahkan lainnya Akun AWS.

   Kotak Akun AWS dialog Lainnya muncul.
8. Di kotak dialog, pilih Tambahkan yang lain Akun AWS. Untuk arn:aws:iam::, masukkan ID akun bucket sumber.
9. Pilih Simpan perubahan.

Untuk memberikan izin kepada pemilik bucket sumber untuk menggunakan kunci KMS (AWS CLI)

- Untuk informasi tentang perintah `put-key-policy` AWS Command Line Interface (AWS CLI), lihat put-key-policydi AWS CLI Command Reference. Untuk informasi tentang operasi API `PutKeyPolicy` yang mendasarinya, lihat PutKeyPolicy di Referensi API AWS Key Management Service.

AWS KMS pertimbangan kuota transaksi

Saat Anda menambahkan banyak objek baru dengan AWS KMS enkripsi setelah mengaktifkan Replikasi Lintas Wilayah (CRR), Anda mungkin mengalami pelambatan (kesalahan HTTP). `503 Service Unavailable` Throttling terjadi ketika jumlah AWS KMS transaksi per detik melebihi kuota saat ini. Untuk informasi selengkapnya, lihat Kuota di Panduan Developer AWS Key Management Service .

Untuk meminta peningkatan kuota, gunakan Kuota Layanan. Untuk informasi selengkapnya, lihat Meminta peningkatan kuota. Jika Service Quotas tidak didukung di Wilayah Anda, buka kasing. AWS Support

Mengaktifkan replikasi untuk objek terenkripsi

Secara default, Amazon S3 tidak mereplikasi objek yang dienkripsi dengan menggunakan enkripsi sisi server dengan () kunci (SSE-KMS) atau enkripsi sisi server AWS Key Management Service dua lapis dengan kunci (DSSE-KMS AWS KMS). AWS KMS Untuk mereplikasi objek yang dienkripsi dengan SSE-KMS atau DSS-KMS, Anda harus memodifikasi konfigurasi replikasi bucket agar Amazon S3 dapat mereplikasi objek tersebut. Contoh ini menjelaskan cara menggunakan konsol Amazon S3 dan AWS Command Line Interface (AWS CLI) untuk mengubah konfigurasi replikasi bucket guna mengaktifkan replikasi objek terenkripsi.

Untuk informasi selengkapnya, lihat Mereplikasi objek terenkripsi (SSE-S3, SSE-KMS, SSE-KMS).

> **Note**
>
> Saat Kunci Bucket S3 diaktifkan untuk bucket sumber atau tujuan, konteks enkripsi akan menjadi Amazon Resource Name (ARN) bucket, bukan ARN objek. Anda harus memperbarui kebijakan IAM Anda untuk menggunakan ARN bucket untuk konteks enkripsi. Untuk informasi selengkapnya, lihat Replikasi dan Kunci Bucket S3.

> **Note**
>
> Anda dapat menggunakan Multi-wilayah AWS KMS keys di Amazon S3. Namun, Amazon S3 saat ini memperlakukan kunci multi-Wilayah selayaknya kunci satu Wilayah, dan tidak menggunakan fitur multi-Wilayah dari kunci tersebut. Untuk informasi selengkapnya, lihat Menggunakan kunci multi-Wilayah di Panduan Pengembang AWS Key Management Service .

Menggunakan konsol S3

Untuk step-by-step instruksi, lihatMengonfigurasi replikasi untuk bucket sumber dan tujuan yang dimiliki oleh akun yang sama. Topik ini memberikan instruksi untuk mengatur konfigurasi replikasi ketika bucket dimiliki oleh yang sama dan berbeda. Akun AWS

Menggunakan AWS CLI

Untuk mereplikasi objek terenkripsi dengan AWS CLI, Anda melakukan hal berikut:

- Buat bucket sumber dan tujuan, lalu aktifkan Penentuan Versi pada bucket ini.
- Buat peran layanan AWS Identity and Access Management (IAM) yang memberikan izin Amazon S3 untuk mereplikasi objek. Izin peran IAM mencakup izin yang diperlukan untuk mereplikasi objek yang dienkripsi.
- Tambahkan konfigurasi replikasi ke bucket sumber. Konfigurasi replikasi menyediakan informasi yang terkait dengan replikasi objek yang dienkripsi dengan menggunakan kunci KMS.
- Tambahkan objek terenkripsi ke bucket sumber.
- Uji penyiapan untuk mengonfirmasi bahwa objek terenkripsi Anda sedang direplikasi ke bucket tujuan.

Prosedur berikut memandu Anda menjalankan proses ini.

Untuk mereplikasi objek yang dienkripsi di sisi server (AWS CLI)

1. Dalam contoh ini, Anda membuat bucket *example-s3-source-bucket* dan *example-s3-destination-bucket* dalam Akun AWS yang sama. Anda juga menetapkan profil kredensial untuk AWS CLI. Contoh ini menggunakan nama profil *acctA*.

   Untuk informasi selengkapnya tentang menyetel profil kredensi, lihat Profil Bernama di Panduan AWS Command Line Interface Pengguna. Untuk menggunakan contoh dalam perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.
2. Gunakan perintah berikut untuk membuat bucket *DOC-EXAMPLE-SOURCE-BUCKET* dan mengaktifkan Penentuan Versi di dalamnya. Contoh perintah berikut membuat bucket *DOC-EXAMPLE-SOURCE-BUCKET* di Wilayah AS Timur (Virginia Utara) (`us-east-1`).

```
aws s3api create-bucket \
--bucket DOC-EXAMPLE-SOURCE-BUCKET \
--region us-east-1 \
```

```
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket DOC-EXAMPLE-SOURCE-BUCKET \
--versioning-configuration Status=Enabled \
--profile acctA
```

3. Gunakan perintah berikut untuk membuat bucket *DOC-EXAMPLE-DESTINATION-BUCKET* dan mengaktifkan Penentuan Versi di dalamnya. Contoh perintah berikut membuat bucket *DOC-EXAMPLE-DESTINATION-BUCKET* di Wilayah AS Barat (Oregon) (`us-west-2`).

> **Note**
>
> Untuk mengatur konfigurasi replikasi saat keduanya *DOC-EXAMPLE-SOURCE-BUCKET*dan *DOC-EXAMPLE-DESTINATION-BUCKET*bucket berada dalam kondisi yang sama Akun AWS, Anda menggunakan profil yang sama. Dalam contoh ini, kami menggunakan *acctA*. Untuk mengonfigurasi replikasi ketika bucket dimiliki oleh Akun AWS yang berbeda, Anda menentukan profil yang berbeda untuk masing-masing bucket.

```
aws s3api create-bucket \
--bucket DOC-EXAMPLE-DESTINATION-BUCKET \
--region us-west-2 \
--create-bucket-configuration LocationConstraint=us-west-2 \
--profile acctA
```

```
aws s3api put-bucket-versioning \
--bucket DOC-EXAMPLE-DESTINATION-BUCKET \
--versioning-configuration Status=Enabled \
--profile acctA
```

4. Di samping itu, Anda membuat peran layanan IAM. Anda akan menentukan peran ini dalam konfigurasi replikasi yang Anda tambahkan ke bucket *DOC-EXAMPLE-SOURCE-BUCKET* nanti. Amazon S3 mengasumsikan peran ini untuk mereplikasi objek atas nama Anda. Anda membuat peran IAM dalam dua langkah:

   - Membuat peran layanan.
   - Lampirkan kebijakan izin pada peran tersebut.

a. Untuk membuat peran layanan IAM, lakukan hal berikut:

   i. Salin kebijakan kepercayaan berikut dan simpan ke file bernama `s3-role-trust-policy-kmsobj.json` di direktori saat ini di komputer lokal Anda. Kebijakan ini memberi Amazon S3 izin pengguna utama layanan untuk mengasumsikan peran sehingga Amazon S3 dapat melakukan tugas atas nama Anda.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

   ii. Gunakan perintah berikut ini untuk membuat peran tersebut:

```
$ aws iam create-role \
--role-name replicationRolekmsobj \
--assume-role-policy-document file://s3-role-trust-policy-kmsobj.json  \
--profile acctA
```

b. Selanjutnya, Anda melampirkan kebijakan izin pada peran tersebut. Kebijakan ini memberikan izin untuk berbagai bucket dan tindakan objek Amazon S3.

   i. Salin kebijakan kepercayaan berikut dan simpan di berkas dengan nama `s3-role-permissions-policykmsobj.json` di direktori saat ini di komputer lokal Anda. Anda akan membuat peran IAM dan melampirkan kebijakannya nanti.

> **Important**
>
> Dalam kebijakan izin, Anda menentukan ID AWS KMS kunci yang akan digunakan untuk enkripsi *`example-s3-source-bucket`* dan *`example-s3-destination-bucket`* bucket. Anda harus membuat dua kunci KMS

terpisah untuk bucket *example-s3-source-bucket* dan *example-s3-destination-bucket*. AWS KMS keys tidak dibagi Wilayah AWS di luar tempat mereka diciptakan.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Action":[
            "s3:ListBucket",
            "s3:GetReplicationConfiguration",
            "s3:GetObjectVersionForReplication",
            "s3:GetObjectVersionAcl",
            "s3:GetObjectVersionTagging"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::example-s3-source-bucket",
            "arn:aws:s3:::example-s3-source-bucket/*"
         ]
      },
      {
         "Action":[
            "s3:ReplicateObject",
            "s3:ReplicateDelete",
            "s3:ReplicateTags"
         ],
         "Effect":"Allow",
         "Condition":{
            "StringLikeIfExists":{
               "s3:x-amz-server-side-encryption":[
                  "aws:kms",
                  "AES256",
                  "aws:kms:dsse"
               ],
               "s3:x-amz-server-side-encryption-aws-kms-key-id":[
                  "AWS KMS key IDs(in ARN format) to use for encrypting
 object replicas"
               ]
            }
         },
```

```
            "Resource":"arn:aws:s3:::example-s3-destination-bucket/*"
        },
        {
            "Action":[
                "kms:Decrypt"
            ],
            "Effect":"Allow",
            "Condition":{
                "StringLike":{
                    "kms:ViaService":"s3.us-east-1.amazonaws.com",
                    "kms:EncryptionContext:aws:s3:arn":[
                        "arn:aws:s3:::example-s3-source-bucket/*"
                    ]
                }
            },
            "Resource":[
                "AWS KMS key IDs(in ARN format) used to encrypt source
 objects."
            ]
        },
        {
            "Action":[
                "kms:Encrypt"
            ],
            "Effect":"Allow",
            "Condition":{
                "StringLike":{
                    "kms:ViaService":"s3.us-west-2.amazonaws.com",
                    "kms:EncryptionContext:aws:s3:arn":[
                        "arn:aws:s3:::example-s3-destination-bucket/*"
                    ]
                }
            },
            "Resource":[
                "AWS KMS key IDs(in ARN format) to use for encrypting object
 replicas"
            ]
        }
    ]
}
```

ii. Buat kebijakan dan lampirkan ke peran tersebut.

```
$ aws iam put-role-policy \
```

```
--role-name replicationRolekmsobj \
--policy-document file://s3-role-permissions-policykmsobj.json \
--policy-name replicationRolechangeownerPolicy \
--profile acctA
```

5. Selanjutnya, tambahkan konfigurasi replikasi berikut ke bucket *example-s3-source-bucket*. Ini memberi tahu Amazon S3 untuk mereplikasi objek dengan awalan `Tax/` ke bucket *example-s3-destination-bucket*.

> **Important**
>
> Dalam konfigurasi replikasi, Anda menentukan peran IAM yang dapat diasumsikan oleh Amazon S3. Anda dapat melakukannya hanya jika Anda memiliki izin `iam:PassRole`. Profil yang Anda tentukan dalam perintah CLI harus memiliki izin ini. Untuk informasi selengkapnya, lihat Memberikan izin pengguna untuk meneruskan peran ke Layanan AWS di Panduan Pengguna IAM.

```
<ReplicationConfiguration>
 <Role>IAM-Role-ARN</Role>
 <Rule>
   <Priority>1</Priority>
   <DeleteMarkerReplication>
      <Status>Disabled</Status>
   </DeleteMarkerReplication>
   <Filter>
      <Prefix>Tax</Prefix>
   </Filter>
   <Status>Enabled</Status>
   <SourceSelectionCriteria>
     <SseKmsEncryptedObjects>
       <Status>Enabled</Status>
     </SseKmsEncryptedObjects>
   </SourceSelectionCriteria>
   <Destination>
     <Bucket>arn:aws:s3:::example-s3-destination-bucket</Bucket>
     <EncryptionConfiguration>
       <ReplicaKmsKeyID>AWS KMS key IDs to use for encrypting object replicas</
ReplicaKmsKeyID>
     </EncryptionConfiguration>
   </Destination>
```

```
  </Rule>
</ReplicationConfiguration>
```

Untuk menambahkan konfigurasi replikasi ke bucket *example-s3-source-bucket*, lakukan hal berikut:

a. Ini AWS CLI mengharuskan Anda untuk menentukan konfigurasi replikasi sebagai JSON. Simpan JSON berikut dalam file (`replication.json`) dalam direktori saat ini di komputer lokal Anda.

```
{
   "Role":"IAM-Role-ARN",
   "Rules":[
      {
         "Status":"Enabled",
         "Priority":1,
         "DeleteMarkerReplication":{
            "Status":"Disabled"
         },
         "Filter":{
            "Prefix":"Tax"
         },
         "Destination":{
            "Bucket":"arn:aws:s3:::example-s3-destination-bucket",
            "EncryptionConfiguration":{
               "ReplicaKmsKeyID":"AWS KMS key IDs (in ARN format) to use for
 encrypting object replicas"
            }
         },
         "SourceSelectionCriteria":{
            "SseKmsEncryptedObjects":{
               "Status":"Enabled"
            }
         }
      }
   ]
}
```

b. Edit JSON guna memberikan nilai untuk bucket *example-s3-destination-bucket*, *AWS KMS key IDs (in ARN format)*, dan *IAM-role-ARN*. Simpan perubahan.

c. Gunakan perintah berikut untuk menambahkan konfigurasi replikasi ke bucket *example-s3-source-bucket* Anda. Pastikan untuk memberikan nama bucket *example-s3-source-bucket*.

```
$ aws s3api put-bucket-replication \
--replication-configuration file://replication.json \
--bucket example-s3-source-bucket \
--profile acctA
```

6. Uji konfigurasi untuk memverifikasi bahwa objek terenkripsi direplikasi. Di konsol Amazon S3, lakukan berikut ini:

   a. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

   b. Di bucket *example-s3-source-bucket*, buat folder bernama `Tax`.

   c. Tambahkan objek sampel ke folder. Pastikan untuk memilih opsi enkripsi dan tentukan kunci KMS Anda untuk mengenkripsi objek.

   d. Pastikan bahwa bucket *example-s3-destination-bucket* berisi replika objek dan dienkripsi menggunakan kunci KMS yang Anda tentukan dalam konfigurasi. Untuk informasi selengkapnya, lihat the section called “Mendapatkan status replikasi”.

Menggunakan AWS SDK

Untuk contoh kode yang menunjukkan cara menambahkan konfigurasi replikasi, lihat Menggunakan AWS SDK. Anda harus memodifikasi konfigurasi replikasi dengan tepat.

Untuk informasi konseptual, lihat Mereplikasi objek terenkripsi (SSE-S3, SSE-KMS, SSE-KMS).

Mereplikasi perubahan metadata dengan sinkronisasi modifikasi replika Amazon S3

Sinkronisasi modifikasi replika Amazon S3 dapat membantu Anda membiarkan metadata objek seperti tag, ACL, dan pengaturan Kunci Objek direplikasi antara replika dan objek sumber. Secara default, Amazon S3 mereplikasi metadata dari objek sumber ke replika saja. Saat sinkronisasi modifikasi replika diaktifkan, Amazon S3 mereplikasi perubahan metadata yang dibuat pada salinan replika kembali ke objek sumber, sehingga membuat replikasi dua arah.

Mengaktifkan sinkronisasi modifikasi replika

Anda dapat menggunakan sinkronisasi modifikasi replika Amazon S3 dengan aturan replikasi baru atau yang sudah ada. Anda dapat menerapkannya ke seluruh bucket S3 atau ke objek Amazon S3 yang memiliki awalan tertentu.

Untuk mengaktifkan sinkronisasi modifikasi replika menggunakan konsol Amazon S3, lihat Contoh untuk mengonfigurasi replikasi langsung. Topik ini memberikan petunjuk untuk mengaktifkan sinkronisasi modifikasi replika dalam konfigurasi replikasi Anda saat bucket dimiliki oleh yang sama atau berbeda. Akun AWS

Untuk mengaktifkan sinkronisasi modifikasi replika menggunakan AWS Command Line Interface (AWS CLI), Anda harus menambahkan konfigurasi replikasi ke bucket yang berisi replika yang diaktifkan. `ReplicaModifications` *Untuk menyiapkan replikasi dua arah, buat aturan replikasi dari bucket sumber (example-s3-bucket1) ke bucket yang berisi replika (contoh-s3-bucket2). Kemudian, buat aturan replikasi kedua dari bucket yang berisi replika (example-s3-bucket2) ke bucket sumber (example-s3-bucket1).* Ember bisa sama, atau berbeda, Wilayah AWS.

> **Note**
>
> Anda harus mengaktifkan sinkronisasi modifikasi replika pada kedua bucket untuk mereplikasi perubahan metadata replika seperti daftar kontrol akses (ACL) objek, tag objek, atau pengaturan Kunci Objek pada objek yang direplikasi. Seperti semua aturan replikasi, aturan ini dapat diterapkan ke seluruh bucket Amazon S3 atau subset objek Amazon S3 yang difilter oleh awalan atau tag objek.

Dalam contoh konfigurasi berikut, Amazon S3 mereplikasi perubahan metadata di bawah awalan *Pajak* ke bucket *example-s3-bucket, yang akan berisi objek sumber*.

```
{
    "Rules": [
        {
            "Status": "Enabled",
            "Filter": {
                "Prefix": "Tax"
            },
            "SourceSelectionCriteria": {
                "ReplicaModifications":{
```

```
                    "Status": "Enabled"
                }
            },
            "Destination": {
                "Bucket": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
            },
            "Priority": 1
        }
    ],
    "Role": "IAM-Role-ARN"
}
```

Untuk petunjuk lengkap tentang membuat aturan replikasi menggunakan AWS CLI, lihatMengonfigurasi replikasi untuk bucket sumber dan tujuan yang dimiliki oleh akun yang sama.

Mereplikasi penanda hapus di antara bucket

Secara default, saat Replikasi S3 diaktifkan dan objek dihapus di bucket sumber, Amazon S3 menambahkan penanda hapus hanya dalam bucket sumber. Ini melindungi data dari penghapusan berbahaya.

Jika Anda memiliki replikasi penanda hapus yang diaktifkan, penanda ini disalin ke bucket tujuan, dan Amazon S3 berperilaku seolah-olah objek tersebut dihapus di bucket sumber dan tujuan. Untuk informasi selengkapnya tentang cara kerja penanda hapus, lihat Bekerja dengan penanda hapus.

> **Note**
>
> Replikasi penanda hapus tidak didukung untuk aturan replikasi berbasis tag. Repllikasi penanda hapus juga tidak mengikuti SLA 15 menit yang diberikan saat menggunakan Kontrol Waktu Replikasi S3.

Jika Anda tidak menggunakan versi konfigurasi replikasi terbaru, operasi penghapusan akan memengaruhi replikasi secara berbeda. Untuk informasi selengkapnya, lihat Bagaimana cara menghapus operasi yang memengaruhi replikasi.

Mengaktifkan replikasi penanda hapus

Anda dapat mulai menggunakan replikasi penanda hapus dengan aturan replikasi baru atau yang sudah ada. Anda dapat menerapkannya ke seluruh bucket S3 atau ke objek Amazon S3 yang memiliki awalan tertentu.

> **Note**
>
> Saat Anda mengaktifkan replikasi penanda hapus dan bucket Anda memiliki aturan kedaluwarsa Siklus Hidup S3, penanda hapus yang ditambahkan oleh aturan kedaluwarsa Siklus Hidup S3 tidak akan direplikasi ke bucket tujuan.

Untuk mengaktifkan replikasi penanda hapus menggunakan konsol Amazon S3, lihat Menggunakan konsol S3. Topik ini memberikan petunjuk untuk mengaktifkan replikasi penanda hapus dalam konfigurasi replikasi Anda saat bucket dimiliki oleh yang sama atau berbeda. Akun AWS

Untuk mengaktifkan replikasi penanda hapus menggunakan AWS Command Line Interface (AWS CLI), Anda harus menambahkan konfigurasi replikasi ke bucket sumber dengan `DeleteMarkerReplication` diaktifkan.

Dalam contoh konfigurasi berikut, penanda hapus direplikasi ke bucket tujuan *DOC-EXAMPLE-BUCKET* untuk objek di bawah awalan *Pajak*.

```
{
    "Rules": [
        {
            "Status": "Enabled",
            "Filter": {
                "Prefix": "Tax"
            },
            "DeleteMarkerReplication": {
                "Status": "Enabled"
            },
            "Destination": {
                "Bucket": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
            },
            "Priority": 1
        }
    ],
    "Role": "IAM-Role-ARN"
}
```

Untuk petunjuk lengkap tentang membuat aturan replikasi melalui AWS CLI, lihat Mengonfigurasi replikasi untuk bucket sumber dan tujuan yang dimiliki oleh akun yang sama di bagian Panduan replikasi.

Mengelola atau menjeda replikasi langsung

Replikasi langsung adalah penyalinan objek otomatis dan asinkron di seluruh ember dalam hal yang sama atau berbeda. Wilayah AWS Setelah menyiapkan konfigurasi replikasi, Amazon S3 mereplikasi objek yang baru dibuat dan pembaruan objek dari bucket sumber ke satu atau beberapa bucket tujuan yang ditentukan.

Anda menggunakan konsol Amazon S3 untuk menambahkan aturan replikasi ke bucket sumber. Aturan replikasi menentukan objek bucket sumber mana yang akan direplikasi dan bucket tujuan atau bucket tempat objek yang direplikasi disimpan. Untuk informasi selengkapnya tentang replikasi, lihat Mereplikasi ikhtisar objek.

Anda dapat mengelola aturan replikasi di halaman Replikasi. Anda dapat menambahkan, melihat, mengaktifkan, menonaktifkan, atau menghapus aturan replikasi. Anda juga dapat mengubah prioritas aturan replikasi Anda. Untuk informasi tentang menambahkan aturan replikasi ke bucket, lihat Menggunakan konsol S3.

Untuk mengelola aturan replikasi bucket S3 dengan menggunakan konsol Amazon S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pada tab Bucket tujuan umum, pilih nama bucket yang Anda inginkan.
4. Pilih tab Manajemen, lalu gulir ke bawah ke Aturan replikasi.
5. Anda dapat mengubah aturan replikasi Anda dengan cara berikut:
   - Untuk mengaktifkan atau menonaktifkan aturan replikasi, pilih tombol opsi di sebelah kiri aturan. Pada menu Tindakan, pilih Aktifkan aturan atau Nonaktifkan aturan. Anda juga dapat menonaktifkan, mengaktifkan, atau menghapus semua aturan di bucket dari menu Tindakan.

> **Note**
>
> Jika Anda menonaktifkan aturan replikasi dan kemudian mengaktifkan kembali aturan tersebut, objek baru atau yang diubah yang tidak direplikasi saat aturan dinonaktifkan tidak akan direplikasi secara otomatis saat aturan diaktifkan kembali. Untuk mereplikasi objek tersebut, Anda harus menggunakan Replikasi Batch S3. Untuk informasi selengkapnya, lihat the section called “Mereplikasi objek yang ada”.

- Untuk mengubah prioritas aturan, pilih tombol opsi di sebelah kiri aturan, lalu pilih Edit aturan.

  Anda menetapkan prioritas aturan untuk menghindari konflik yang disebabkan oleh objek yang termasuk dalam cakupan lebih dari satu aturan. Dalam kasus aturan yang tumpang tindih, Amazon S3 menggunakan prioritas aturan untuk menentukan aturan mana yang berlaku. Semakin tinggi angkanya, semakin tinggi prioritasnya. Untuk informasi selengkapnya tentang prioritas aturan, lihat Konfigurasi Replikasi.

Menjeda atau menghentikan replikasi

Untuk menjeda sementara replikasi dan membuatnya dilanjutkan secara otomatis nanti, Anda dapat menggunakan `aws:s3:bucket-pause-replication` tindakan di. AWS Fault Injection ServiceUntuk informasi selengkapnya, lihat aws:s3:bucket-pause-replicationdan Jeda Replikasi S3 di Panduan Pengguna.AWS Fault Injection Service

Untuk menghentikan replikasi di Amazon S3, sebaiknya nonaktifkan aturan replikasi Anda. Jika Anda menonaktifkan aturan replikasi dan kemudian mengaktifkan kembali aturan tersebut, objek baru atau yang diubah yang tidak direplikasi saat aturan dinonaktifkan tidak akan direplikasi secara otomatis saat aturan diaktifkan kembali. Untuk mereplikasi objek tersebut, Anda harus menggunakan Replikasi Batch S3. Untuk informasi selengkapnya, lihat the section called “Mereplikasi objek yang ada”.

Replikasi juga akan berhenti jika Anda menghapus peran AWS Identity and Access Management (IAM), izin AWS Key Management Service (AWS KMS), atau izin kebijakan bucket yang memberikan Amazon S3 izin yang diperlukan. Namun, kami tidak merekomendasikan pendekatan ini karena menyebabkan replikasi gagal. Amazon S3 melaporkan status replikasi objek yang terdampak sebagai `FAILED`. Jika izin kemudian dipulihkan, objek yang ditandai sebagai `FAILED` tidak direplikasi secara otomatis. Untuk mereplikasi objek tersebut, Anda harus menggunakan Replikasi Batch S3.

Memantau progres dengan metrik replikasi dan Notifikasi Peristiwa S3

Metrik Replikasi S3 memberikan metrik terperinci untuk aturan replikasi dalam konfigurasi replikasi Anda. Dengan metrik replikasi, Anda dapat memantau minute-by-minute kemajuan dengan melacak byte yang tertunda, operasi tertunda, operasi yang gagal replikasi, dan latensi replikasi.

Metrik Replikasi S3 diaktifkan secara otomatis ketika Anda mengaktifkan Kontrol Waktu Replikasi S3 (S3 RTC). Anda juga dapat mengaktifkan metrik Replikasi S3 secara terpisah dari S3 RTC saat membuat atau mengedit aturan. S3 RTC mencakup fitur lain seperti perjanjian tingkat layanan (SLA)

dan pemberitahuan atas ambang batas yang terlewat. Untuk informasi selengkapnya, lihat Memenuhi persyaratan kepatuhan menggunakan Kontrol Waktu Replikasi S3 (S3 RTC).

Metrik byte tertunda, operasi tertunda, dan latensi replikasi hanya berlaku untuk objek baru yang direplikasi dengan Replikasi Lintas-Wilayah S3 (S3 CRR) atau Replikasi Wilayah Sama S3 (S3 SRR). Metrik operasi yang mereplikasi melacak kedua objek baru yang direplikasi dengan S3 CRR atau S3 SRR dan objek lama yang direplikasi dengan Replikasi Batch S3. Untuk membantu memecahkan masalah konfigurasi, Anda juga dapat menyiapkan Notifikasi Peristiwa Amazon S3 untuk menerima peristiwa kegagalan replikasi.

Saat diaktifkan, metrik Replikasi S3 mempublikasikan metrik berikut ke Amazon: CloudWatch

- Byte dengan Replikasi Tertunda–Jumlah total byte objek yang menunggu replikasi untuk aturan replikasi yang ditentukan.
- Latensi Replikasi–Jumlah detik maksimum saat bucket tujuan replikasi berada di belakang bucket sumber untuk aturan replikasi tertentu.
- Operasi Replikasi Tertunda–Jumlah operasi yang menunggu replikasi untuk aturan replikasi yang ditentukan. Metrik ini melacak operasi yang terkait dengan objek, penanda hapus, tag, daftar kontrol akses (ACL), dan Kunci Objek S3.
- Operasi Replikasi Gagal–Jumlah operasi yang gagal mereplikasi untuk aturan replikasi yang ditentukan. Metrik ini melacak operasi yang terkait dengan objek, penanda hapus, tag, ACL, dan Kunci Objek. Berbeda dengan metrik replikasi lainnya, metrik ini berlaku baik untuk objek baru yang direplikasi dengan S3 CRR atau S3 SRR dan untuk objek lama yang direplikasi dengan Replikasi Batch S3.

> **Note**
>
> Operasi Replikasi Gagal melacak kegagalan Replikasi S3 yang dikumpulkan pada interval per menit. Untuk mengidentifikasi objek tertentu yang gagal mereplikasi dan alasan kegagalannya, berlangganan peristiwa `OperationFailedReplication` di Notifikasi Peristiwa Amazon S3. Untuk informasi selengkapnya, lihat Menerima peristiwa kegagalan replikasi dengan Notifikasi Peristiwa Amazon S3.

Jika pekerjaan gagal dijalankan sama sekali, metrik tidak dikirim ke Amazon CloudWatch. Misalnya, tugas Anda tidak akan berjalan jika Anda tidak memiliki izin yang diperlukan untuk menjalankan tugas Replikasi Batch S3, atau jika tag atau awalan dalam konfigurasi replikasi tidak cocok.

Topik

- Mengaktifkan metrik Replikasi S3
- Menerima peristiwa kegagalan replikasi dengan Notifikasi Peristiwa Amazon S3
- Melihat metrik replikasi dengan Lensa Penyimpanan S3
- Melihat metrik replikasi dengan menggunakan konsol Amazon S3
- Penyebab kegagalan replikasi Amazon S3
- Mendapatkan informasi status replikasi

Mengaktifkan metrik Replikasi S3

Anda dapat mulai menggunakan metrik Replikasi S3 dengan aturan replikasi baru atau yang sudah ada. Anda dapat memilih untuk menerapkan aturan replikasi Anda ke seluruh bucket S3, atau ke objek Amazon S3 dengan awalan atau tag tertentu.

Topik ini memberikan petunjuk untuk mengaktifkan metrik Replikasi S3 dalam konfigurasi replikasi Anda saat bucket sumber dan tujuan dimiliki oleh Akun AWS yang sama atau berbeda.

Untuk mengaktifkan metrik replikasi menggunakan AWS Command Line Interface (AWS CLI), Anda harus menambahkan konfigurasi replikasi ke bucket sumber dengan diaktifkan. `Metrics` Dalam konfigurasi contoh ini, objek di bawah awalan *`Tax`* direplikasi ke bucket tujuan *`DOC-EXAMPLE-BUCKET`*, dan metrik dibuat untuk objek tersebut.

```
{
    "Rules": [
        {
            "Status": "Enabled",
            "Filter": {
                "Prefix": "Tax"
            },
            "Destination": {
                "Bucket": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "Metrics": {
                    "Status": "Enabled"
                }
            },
            "Priority": 1
        }
    ],
    "Role": "IAM-Role-ARN"
```

```
}
```

Untuk petunjuk lengkap tentang pembuatan aturan replikasi, lihat Mengonfigurasi replikasi untuk bucket sumber dan tujuan yang dimiliki oleh akun yang sama.

Untuk informasi selengkapnya tentang melihat metrik replikasi di konsol S3, lihat Melihat metrik replikasi dengan menggunakan konsol Amazon S3.

> **Note**
>
> Metrik Replikasi S3 ditagih dengan tarif yang sama dengan metrik kustom Amazon. CloudWatch Untuk informasi lebih lanjut, lihat harga Amazon CloudWatch.

Menerima peristiwa kegagalan replikasi dengan Notifikasi Peristiwa Amazon S3

Notifikasi Peristiwa S3 dapat memberi tahu Anda dalam instans ketika objek tidak mereplikasi ke tujuannya Wilayah AWS. Acara Amazon S3 tersedia melalui Amazon Simple Queue Service (Amazon SQS), Amazon Simple Notification Service (Amazon SNS), atau. AWS Lambda Untuk informasi selengkapnya, lihat the section called “Notifikasi Peristiwa Amazon S3”.

Untuk daftar kode kegagalan yang diambil oleh Notifikasi Peristiwa S3, lihat Penyebab kegagalan replikasi Amazon S3.

Melihat metrik replikasi dengan Lensa Penyimpanan S3

Guna mendapatkan metrik terperinci untuk Replikasi S3, termasuk metrik jumlah aturan replikasi, Anda dapat menggunakan Lensa Penyimpanan Amazon S3. Lensa Penyimpanan S3 adalah fitur analisis penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan objek. Untuk informasi selengkapnya, lihat Menggunakan Lensa Penyimpanan S3 untuk melindungi data Anda. Untuk daftar metrik lengkap, lihat glosarium metrik Lensa Penyimpanan S3.

Melihat metrik replikasi dengan menggunakan konsol Amazon S3

Ada tiga jenis metrik Amazon untuk Amazon S3: CloudWatch metrik penyimpanan, metrik permintaan, dan metrik replikasi. Metrik Replikasi S3 diaktifkan secara otomatis saat Anda mengaktifkan replikasi dengan Kontrol Waktu Replikasi S3 (S3 RTC) dengan menggunakan atau Amazon S3 API. AWS Management Console Anda juga dapat mengaktifkan metrik Replikasi S3 secara terpisah dari S3 RTC saat membuat atau mengedit aturan.

Metrik replikasi melacak ID aturan dari konfigurasi replikasi. ID aturan replikasi dapat dikhususkan untuk awalan, tag, atau kombinasi keduanya.

Untuk informasi selengkapnya tentang CloudWatch metrik untuk Amazon S3, lihat. Memantau metrik dengan Amazon CloudWatch

Prasyarat

Aktifkan aturan replikasi yang memiliki metrik Replikasi S3.

Untuk melihat metrik replikasi

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket. Di daftar Bucket, pilih nama bucket berisi objek yang ingin Anda lihat metrik replikasinya.
3. Pilih tab Metrik.
4. Di bawah Metrik replikasi, pilih Aturan replikasi.
5. Pilih Tampilkan bagan.

   Amazon S3 menampilkan bagan Latensi Replikasi (dalam detik), Byte dengan replikasi tertunda, Operasi replikasi tertunda, dan Operasi replikasi gagal.

Anda kemudian dapat melihat metrik replikasi Latensi replikasi (dalam detik), Operasi replikasi tertunda, Byte dengan replikasi tertunda, dan Operasi replikasi gagal untuk aturan yang Anda pilih. Jika Anda menggunakan Kontrol Waktu Replikasi S3, Amazon CloudWatch mulai melaporkan metrik replikasi 15 menit setelah Anda mengaktifkan S3 RTC pada aturan replikasi masing-masing. Anda dapat melihat metrik replikasi di konsol Amazon S3 atau konsol. CloudWatch Untuk informasi selengkapnya, lihat Metrik replikasi dengan RTC S3.

> **Note**
>
> Anda juga dapat melihat metrik terperinci untuk Replikasi S3 di konsol Amazon S3 dengan menggunakan Lensa Penyimpanan Amazon S3. Lensa Penyimpanan S3 adalah fitur analisis penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan objek. Untuk informasi selengkapnya, lihat Menggunakan Lensa Penyimpanan S3 untuk melindungi data Anda. Untuk daftar metrik lengkap, lihat glosarium metrik Lensa Penyimpanan S3.

Penyebab kegagalan replikasi Amazon S3

Tabel berikut mencantumkan alasan kegagalan replikasi Amazon S3. Anda dapat melihat alasan ini dengan menerima peristiwa failureReason dengan Notifikasi Peristiwa Amazon S3. Anda dapat menerima Pemberitahuan Acara S3 melalui Amazon Simple Queue Service (Amazon Simple Service), Amazon Simple Notification Service (Amazon SNS), atau. AWS Lambda Untuk informasi selengkapnya, lihat Notifikasi Peristiwa Amazon S3.

Anda juga dapat melihat alasan kegagalan ini dalam laporan penyelesaian Replikasi Batch S3. Untuk informasi selengkapnya, lihat Laporan penyelesaian Replikasi Batch.

Penyebab kegagalan replikasi	Deskripsi
`AssumeRoleNotPermitted`	Amazon S3 tidak dapat mengasumsikan peran AWS Identity and Access Managemen t (IAM) yang ditentukan dalam konfigurasi replikasi atau dalam tugas Operasi Batch.
`DstBucketInvalidRegion`	Bucket tujuan tidak Wilayah AWS sama dengan yang ditentukan oleh pekerjaan Operasi Batch. Kesalahan ini khusus untuk Replikasi Batch.
`DstBucketNotFound`	Amazon S3 tidak dapat menemukan bucket tujuan yang ditentukan dalam konfigurasi replikasi.
`DstBucketObjectLockConfigMissing`	Untuk mereplikasi objek dari bucket sumber dengan dukungan Kunci Objek, bucket tujuan juga harus memiliki Kunci Objek yang diaktifkan. Kesalahan ini menunjukkan bahwa Kunci Objek mungkin tidak diaktifkan di bucket tujuan. Untuk informasi selengkapnya, lihat Pertimbangan Kunci Objek.
`DstBucketUnversioned`	Penentuan Versi tidak diaktifkan untuk bucket tujuan S3. Untuk mereplikasi objek dengan

Penyebab kegagalan replikasi	Deskripsi
	Replikasi S3, aktifkan Penentuan Versi untuk bucket tujuan.
`DstDelObjNotPermitted`	Amazon S3 tidak dapat mereplikasi penanda hapus ke bucket tujuan. Izin `s3:ReplicateDelete` mungkin hilang untuk bucket tujuan.
`DstKmsKeyInvalidState`	Kunci AWS Key Management Service (AWS KMS) untuk bucket tujuan tidak dalam status valid. Tinjau dan aktifkan AWS KMS kunci yang diperlukan. Untuk informasi selengkapnya tentang mengelola AWS KMS kunci, lihat Status AWS KMS kunci kunci di Panduan AWS Key Management Service Pengembang.
`DstKmsKeyNotFound`	AWS KMS Kunci yang dikonfigurasi untuk bucket tujuan dalam konfigurasi replikasi tidak ada.
`DstMultipartCompleteNotPermitted`	Amazon S3 tidak dapat menyelesaikan unggahan objek multibagian di bucket tujuan. Izin `s3:ReplicateObject` mungkin tidak ada untuk bucket tujuan.
`DstMultipartInitNotPermitted`	Amazon S3 tidak dapat memulai unggahan objek multibagian ke bucket tujuan. Izin `s3:ReplicateObject` mungkin tidak ada untuk bucket tujuan.
`DstMultipartPartUploadNotPermitted`	Amazon S3 tidak dapat mengunggah objek multibagian ke bucket tujuan. Izin `s3:ReplicateObject` mungkin tidak ada untuk bucket tujuan.

Penyebab kegagalan replikasi	Deskripsi
`DstObjectHardDeleted`	Replikasi Batch S3 tidak mendukung replikasi ulang objek yang dihapus dengan ID versi objek dari bucket tujuan. Kesalahan ini khusus untuk Replikasi Batch.
`DstPutAclNotPermitted`	Amazon S3 tidak dapat mereplikasi daftar kontrol akses (ACL) objek ke bucket tujuan. Izin `s3:ReplicateObject` mungkin tidak ada untuk bucket tujuan.
`DstPutLegalHoldNotPermitted`	Amazon S3 tidak dapat menempatkan penahanan hukum Kunci Objek pada objek tujuan saat mereplikasi objek yang tidak dapat diubah. Izin `s3:PutObjectLegalHold` mungkin hilang untuk bucket tujuan. Untuk informasi selengkapnya, lihat Penahanan legal.
`DstPutObjectNotPermitted`	Amazon S3 tidak dapat mereplikasi objek ke bucket tujuan. Izin `s3:ReplicateObject` atau `s3:ObjectOwnerOverrideToBucketOwner` mungkin hilang untuk bucket tujuan.
`DstPutTaggingNotPermitted`	Amazon S3 tidak dapat mereplikasi tag objek ke bucket tujuan. Izin `s3:ReplicateObject` mungkin hilang untuk bucket tujuan.
`DstVersionNotFound`	Amazon S3 tidak dapat menemukan versi objek yang diperlukan di bucket tujuan yang metadatanya perlu direplikasi.

Penyebab kegagalan replikasi	Deskripsi
`InitiateReplicationNotPermitted`	Amazon S3 tidak dapat memulai replikasi pada objek. Izin `s3:InitiateReplication` mungkin hilang untuk tugas Operasi Batch. Kesalahan ini khusus untuk Replikasi Batch.
`SrcBucketInvalidRegion`	Bucket sumber tidak Wilayah AWS sama dengan yang ditentukan oleh pekerjaan Operasi Batch. Kesalahan ini khusus untuk Replikasi Batch.
`SrcBucketNotFound`	Amazon S3 tidak dapat menemukan bucket sumber.
`SrcBucketReplicationConfigMissing`	Amazon S3 tidak dapat menemukan konfigurasi replikasi untuk bucket sumber.
`SrcGetAclNotPermitted`	Amazon S3 tidak dapat mengakses objek di bucket sumber untuk replikasi. Izin `s3:GetObjectVersionAcl` mungkin hilang untuk objek bucket sumber. Objek di bucket sumber harus dimiliki oleh pemilik bucket. Jika ACL diaktifkan, pastikan apakah Kepemilikan Objek ditetapkan ke Pemilik bucket pilihan atau Penulis objek. Jika Kepemilikan Objek ditetapkan ke Pemilik bucket pilihan, objek bucket sumber harus memiliki ACL `bucket-owner-full-control` agar pemilik bucket menjadi pemilik objek. Akun sumber dapat mengambil kepemilikan semua objek di bucket mereka dengan menetapkan Kepemilikan Objek ke Pemilik bucket yang diberlakukan dan menonaktifkan ACL.

Penyebab kegagalan replikasi	Deskripsi
`SrcGetLegalHoldNotPermitted`	Amazon S3 tidak dapat mengakses informasi penahanan hukum Kunci Objek S3.
`SrcGetObjectNotPermitted`	Amazon S3 tidak dapat mengakses objek di bucket sumber untuk replikasi. Izin `s3:GetObjectVersionForReplication` mungkin tidak ada untuk bucket sumber.
`SrcGetRetentionNotPermitted`	Amazon S3 tidak dapat mengakses informasi periode retensi Kunci Objek S3.
`SrcGetTaggingNotPermitted`	Amazon S3 tidak dapat mengakses informasi tag objek dari bucket sumber. Izin `s3:GetObjectVersionTagging` mungkin hilang untuk bucket sumber.
`SrcHeadObjectNotPermitted`	Amazon S3 tidak dapat mengambil metadata objek dari bucket sumber. Izin `s3:GetObjectVersionForReplication` mungkin hilang untuk bucket sumber.
`SrcKeyNotFound`	Amazon S3 tidak dapat menemukan kunci objek sumber yang akan direplikasi. Objek sumber mungkin telah dihapus sebelum replikasi selesai.
`SrcKmsKeyInvalidState`	AWS KMS Kunci untuk bucket sumber tidak dalam status valid. Tinjau dan aktifkan AWS KMS kunci yang diperlukan. Untuk informasi selengkapnya tentang mengelola AWS KMS kunci, lihat Status AWS KMS kunci kunci di Panduan AWS Key Management Service Pengembang.

Penyebab kegagalan replikasi	Deskripsi
`SrcObjectNotEligible`	Beberapa objek tidak memenuhi syarat untuk replikasi. Ini mungkin karena kelas penyimpanan objek atau tag objek tidak cocok dengan konfigurasi replikasi.
`SrcObjectNotFound`	Objek sumber tidak ada.
`SrcReplicationNotPending`	Amazon S3 telah mereplikasi objek ini. Objek ini tidak lagi menunggu replikasi.
`SrcVersionNotFound`	Amazon S3 tidak dapat menemukan versi objek sumber yang akan direplikasi. Versi objek sumber mungkin telah dihapus sebelum replikasi selesai.

Topik terkait

Menyiapkan izin untuk replikasi langsung

Memecahkan masalah replikasi

Mendapatkan informasi status replikasi

Status replikasi dapat membantu Anda menentukan status saat ini dari objek yang sedang direplikasi. Status replikasi dari objek sumber akan kembali sebagai `PENDING`, `COMPLETED`, atau `FAILED`. Status replikasi dari replika akan mengembalikan `REPLICA`.

Topik

- Gambaran status replikasi
- Status replikasi jika mereplikasi ke beberapa bucket tujuan
- Status replikasi jika sinkronisasi modifikasi replika Amazon S3 diaktifkan
- Menemukan status replikasi

Gambaran status replikasi

Dalam replikasi, Anda memiliki bucket sumber tempat Anda mengonfigurasi replikasi dan tujuan tempat Amazon S3 mereplikasi objek. Saat Anda meminta sebuah objek (menggunakan objek `GET`) atau metadata objek (menggunakan objek `HEAD`) dari bucket ini, Amazon S3 mengembalikan header `x-amz-replication-status` di respons:

- Saat Anda meminta objek dari bucket sumber, Amazon S3 mengembalikan header `x-amz-replication-status` jika objek di permintaan Anda memenuhi syarat untuk replikasi.

  Misalnya, Anda menetapkan awalan objek `TaxDocs` dalam konfigurasi replikasi Anda untuk memberi tahu Amazon S3 agar hanya mereplikasi objek dengan awalan nama kunci `TaxDocs`. Objek apa pun yang Anda unggah yang memiliki awalan nama kunci ini—misalnya, `TaxDocs/document1.pdf`—akan direplikasi. Untuk permintaan objek dengan awalan nama kunci ini, Amazon S3 mengembalikan header `x-amz-replication-status` dengan salah satu nilai berikut untuk status replikasi objek: `PENDING`, `COMPLETED`, atau `FAILED`.

  > **Note**
  >
  > Jika replikasi objek gagal setelah Anda mengunggah sebuah objek, Anda tidak dapat mencoba ulang replikasi. Anda harus mengunggah objek lagi. Transisi objek ke status `FAILED` untuk masalah seperti tidak adanya izin peran replikasi, izin AWS KMS , atau izin bucket. Untuk kegagalan sementara, seperti jika bucket atau Wilayah tidak tersedia, status replikasi tidak akan berpindah ke `FAILED`, tetapi akan tetap `PENDING`. Setelah sumber daya kembali online, S3 akan melanjutkan replikasi objek tersebut.

- Saat Anda meminta objek dari bucket tujuan, jika objek dalam permintaan Anda adalah replika yang dibuat Amazon S3, Amazon S3 mengembalikan header `x-amz-replication-status` dengan nilai `REPLICA`.

> **Note**
>
> Sebelum menghapus sebuah objek dari bucket sumber dengan replikasi yang diaktifkan, periksa status replikasi objek untuk memastikan bahwa objek tersebut telah direplikasi. Jika konfigurasi siklus hidup diaktifkan pada bucket sumber, Amazon S3 menangguhkan tindakan siklus hidup hingga memberi tanda status objek sebagai `COMPLETED` atau `FAILED`.

Status replikasi jika mereplikasi ke beberapa bucket tujuan

Saat Anda mereplikasi objek ke beberapa bucket tujuan, header `x-amz-replication-status` bertindak secara berbeda. Header objek sumber hanya mengembalikan nilai `COMPLETED` saat replikasi berhasil ke semua tujuan. Header tetap pada nilai `PENDING` sampai replikasi telah selesai untuk semua tujuan. Jika satu atau beberapa tujuan gagal replikasi, header mengembalikan `FAILED`.

Status replikasi jika sinkronisasi modifikasi replika Amazon S3 diaktifkan

Saat aturan replikasi Anda mengaktifkan sinkronisasi modifikasi replika Amazon S3, replika dapat melaporkan status selain `REPLICA`. Jika perubahan metadata sedang dalam proses replikasi, header `x-amz-replication-status` mengembalikan `PENDING`. Jika sinkronisasi modifikasi replika gagal mereplikasi metadata, header akan mengembalikan `FAILED`. Jika metadata direplikasi dengan benar, replika mengembalikan header `REPLICA`.

Menemukan status replikasi

Untuk mendapatkan status replikasi dari objek di dalam bucket, Anda dapat menggunakan alat Inventaris Amazon S3. Amazon S3 mengirim file CSV ke bucket tujuan yang Anda tentukan dalam konfigurasi persediaan. Anda juga dapat menggunakan Amazon Athena untuk memerika status replikasi dalam laporan persediaan. Untuk informasi selengkapnya tentang Inventaris Amazon S3, lihat Inventaris Amazon S3.

Anda juga dapat menemukan status replikasi objek menggunakan konsol, AWS Command Line Interface (AWS CLI), atau AWS SDK.

Menggunakan konsol S3

Di konsol S3, Anda dapat melihat status replikasi untuk objek di halaman Detail objek di bawah Tinjauan manajemen objek.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Bucket, pilih nama bucket.
3. Di daftar Objek, pilih nama objek.
4. Di bawah tab Properti temukan Tinjauan manajemen objek, di sini Anda dapat melihat Status replikasi.

Menggunakan AWS CLI

Gunakan perintah `head-object` untuk mengambil metadata objek, seperti berikut.

```
aws s3api head-object --bucket source-bucket --key object-key --version-id object-
version-id
```

Perintah mengembalikan metadata objek, termasuk `ReplicationStatus` seperti yang ditunjukkan dalam contoh tanggapan berikut.

```
{
   "AcceptRanges":"bytes",
   "ContentType":"image/jpeg",
   "LastModified":"Mon, 23 Mar 2015 21:02:29 GMT",
   "ContentLength":3191,
   "ReplicationStatus":"COMPLETED",
   "VersionId":"jfnW.HIMOfYiD_9rGbSkmroXsFj3fqZ.",
   "ETag":"\"6805f2cfc46c0f04559748bb039d69ae\"",
   "Metadata":{

   }
}
```

Menggunakan AWS SDK

Fragmen kode berikut mendapatkan status replikasi dengan AWS SDK for Java dan AWS SDK for .NET, masing-masing.

Java

```
GetObjectMetadataRequest metadataRequest = new GetObjectMetadataRequest(bucketName,
 key);
ObjectMetadata metadata = s3Client.getObjectMetadata(metadataRequest);

System.out.println("Replication Status : " +
 metadata.getRawMetadataValue(Headers.OBJECT_REPLICATION_STATUS));
```

.NET

```
GetObjectMetadataRequest getmetadataRequest = new GetObjectMetadataRequest
    {
```

```
        BucketName = sourceBucket,
        Key        = objectKey
    };

GetObjectMetadataResponse getmetadataResponse =
 client.GetObjectMetadata(getmetadataRequest);
Console.WriteLine("Object replication status: {0}",
 getmetadataResponse.ReplicationStatus);
```

Mereplikasi objek yang ada dengan Replikasi Batch S3

Dengan menggunakan Replikasi Batch S3, Anda dapat mereplikasi jenis objek berikut:

- Objek yang ada sebelum konfigurasi replikasi ada
- Objek yang sebelumnya telah direplikasi
- Objek yang gagal replikasi

Anda dapat mereplikasi objek ini sesuai permintaan dengan menggunakan pekerjaan Operasi Batch. Replikasi Batch S3 berbeda dari replikasi langsung, yang secara terus menerus dan otomatis mereplikasi objek baru di seluruh bucket Amazon S3.

Untuk memulai dengan Batch Replication, Anda dapat:

- Memulai Replikasi Batch untuk aturan atau tujuan replikasi baru — Anda dapat membuat tugas Replikasi Batch satu kali saat membuat aturan pertama dalam konfigurasi replikasi baru atau saat menambahkan tujuan baru ke konfigurasi yang ada melalui konsol Amazon S3.
- Memulai Replikasi Batch untuk konfigurasi replikasi yang ada — Anda dapat membuat pekerjaan Replikasi Batch baru dengan menggunakan Operasi Batch S3 melalui konsol Amazon S3, AWS Command Line Interface (AWS CLI), SDK AWS , atau Amazon S3 REST API.

Ketika tugas Replikasi Batch selesai, Anda menerima laporan penyelesaian. Untuk informasi selengkapnya tentang cara menggunakan laporan untuk memeriksa tugas, lihat Melacak status tugas dan laporan penyelesaian.

Pertimbangan Replikasi Batch S3

- Bucket sumber Anda harus memiliki konfigurasi replikasi yang sudah ada. Untuk mengaktifkan replikasi, lihat Menyiapkan replikasi langsung dan Contoh untuk mengonfigurasi replikasi langsung.

- Jika Siklus Hidup S3 dikonfigurasi untuk bucket, sebaiknya nonaktifkan aturan siklus hidup Anda saat tugas Replikasi Batch aktif. Melakukannya membantu memastikan kesetaraan antara ember sumber dan tujuan. Jika tidak, bucket ini bisa menyimpang, dan bucket tujuan tidak akan menjadi replika yang tepat dari bucket sumber. Sebagai contoh, pertimbangkan alur perencanaan berikut ini:
  - Bucket sumber Anda memiliki beberapa versi objek dan penanda hapus pada objek tersebut.
  - Bucket sumber dan tujuan Anda memiliki konfigurasi siklus hidup untuk menghapus penanda hapus yang kedaluwarsa.

  Dalam skenario ini, Replikasi Batch mungkin mereplikasi penanda hapus ke bucket tujuan sebelum mereplikasi versi objek. Perilaku ini dapat mengakibatkan konfigurasi siklus hidup Anda menandai penanda hapus sebagai kedaluwarsa dan penanda hapus dihapus dari keranjang tujuan sebelum versi objek direplikasi.
- Peran AWS Identity and Access Management (IAM) yang Anda tentukan untuk menjalankan pekerjaan Operasi Batch harus memiliki izin yang diperlukan untuk melakukan operasi Replikasi Batch yang mendasarinya. Untuk informasi selengkapnya tentang cara membuat peran IAM, lihat Mengonfigurasi kebijakan IAM untuk Replikasi Batch.
- Replikasi Batch memerlukan manifes, yang dapat dihasilkan oleh Amazon S3. Manifes yang dihasilkan harus disimpan Wilayah AWS sama dengan bucket sumber. Jika Anda memilih untuk tidak membuat manifes, Anda dapat menyediakan laporan Inventaris Amazon S3 atau file CSV yang berisi objek yang ingin direplikasi.
- Batch Replication tidak mendukung replikasi ulang objek yang dihapus dengan ID versi objek dari bucket tujuan. Untuk mereplikasi ulang objek ini, Anda dapat menyalin objek sumber di tempat dengan tugas Penyalinan Batch. Menyalin objek tersebut di tempat akan membuat versi baru objek di bucket sumber dan secara otomatis memulai replikasi ke bucket tujuan. Menghapus dan membuat ulang bucket tujuan tidak memulai replikasi.

  Untuk informasi selengkapnya tentang Batch Copy, lihatContoh yang menggunakan Operasi Batch untuk menyalin objek.
- Jika Anda menggunakan aturan replikasi pada bucket S3, pastikan untuk memperbarui konfigurasi replikasi Anda dengan memberikan peran IAM yang dilampirkan pada aturan replikasi izin yang tepat untuk mereplikasi objek. Peran IAM ini harus memiliki izin yang diperlukan untuk melakukan replikasi pada bucket sumber dan tujuan.
- Jika Anda mengirimkan beberapa pekerjaan Replikasi Batch untuk bucket yang sama dalam jangka waktu singkat, Amazon S3 akan menjalankan pekerjaan tersebut secara bersamaan.

- Jika Anda mengirimkan beberapa pekerjaan Replikasi Batch untuk dua bucket yang berbeda, ketahuilah bahwa Amazon S3 mungkin tidak menjalankan semua pekerjaan secara bersamaan. Jika Anda melebihi jumlah pekerjaan Replikasi Batch yang dapat berjalan pada satu waktu di akun Anda, Amazon S3 akan menjeda pekerjaan dengan prioritas lebih rendah untuk mengerjakan pekerjaan dengan prioritas lebih tinggi. Setelah item prioritas yang lebih tinggi selesai, pekerjaan yang dijeda akan menjadi aktif kembali.
- Batch Replication tidak didukung untuk objek yang disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive.
- Untuk mereplikasi objek S3 Intelligent-Tiering yang disimpan di tingkat penyimpanan Akses Arsip atau Akses Arsip Dalam, Anda harus terlebih dahulu memulai permintaan pemulihan dan menunggu hingga objek dipindahkan ke tingkat Akses Sering.

Menentukan manifes untuk tugas Replikasi Batch

Manifes adalah objek Amazon S3 yang berisi kunci objek yang harus ditindaklanjuti Amazon S3. Jika Anda ingin membuat tugas Replikasi Batch, Anda harus menyediakan manifes buatan pengguna atau meminta Amazon S3 menghasilkan manifes berdasarkan konfigurasi replikasi Anda.

Jika Anda menyediakan manifes buatan pengguna, manifes tersebut harus dalam bentuk laporan Inventaris Amazon S3 atau file CSV. Jika objek dalam manifes Anda berada dalam bucket berversi, Anda harus menentukan ID versi untuk objek tersebut. Hanya objek dengan ID versi yang ditentukan dalam manifes yang akan direplikasi. Untuk mempelajari penentuan manifes lebih lanjut, lihat Menentukan manifes.

Jika Anda memilih agar Amazon S3 menghasilkan file manifes atas nama Anda, objek yang terdaftar akan menggunakan bucket sumber, awalan, dan tag yang sama dengan semua konfigurasi replikasi bucket sumber. Dengan manifes yang dihasilkan, Amazon S3 akan mereplikasi semua versi objek yang memenuhi syarat.

> **Note**
>
> Jika Anda memilih agar Amazon S3 menghasilkan manifes, manifes harus disimpan Wilayah AWS sama dengan bucket sumber.

Filter untuk tugas Replikasi Batch

Saat membuat pekerjaan Replikasi Batch, Anda dapat secara opsional menentukan filter tambahan, seperti tanggal pembuatan objek dan status replikasi, untuk mengurangi cakupan pekerjaan.

Anda dapat memfilter objek yang akan mereplikasi berdasarkan nilai `ObjectReplicationStatuses`, dengan memberikan satu atau beberapa nilai berikut:

- `"NONE"`–Menunjukkan bahwa Amazon S3 belum pernah mencoba mereplikasi objek sebelumnya.
- `"FAILED"`— Menunjukkan bahwa Amazon S3 telah mencoba, tetapi gagal, untuk mereplikasi objek sebelumnya.
- `"COMPLETED"`–Menunjukkan bahwa Amazon S3 telah berhasil mereplikasi objek sebelumnya.
- `"REPLICA"`— Menunjukkan bahwa ini adalah objek replika yang telah direplikasi Amazon S3 dari sumber lain.

Untuk informasi selengkapnya tentang status replikasi, lihat Mendapatkan informasi status replikasi.

Jika Anda tidak memfilter tugas Replikasi Batch Anda, Operasi Batch akan mencoba untuk mereplikasi semua objek (tidak peduli mereka`ObjectReplicationStatus`) dalam manifes Anda yang cocok dengan aturan dalam konfigurasi replikasi Anda, kecuali untuk objek tertentu yang tidak direplikasi secara default. Untuk informasi selengkapnya, lihat the section called “Apa yang tidak direplikasi dengan konfigurasi replikasi?”

Bergantung pada tujuan Anda, Anda dapat menetapkan `ObjectReplicationStatuses` ke satu atau beberapa nilai berikut:

- Untuk mereplikasi hanya objek yang ada yang belum pernah direplikasi, hanya sertakan. `"NONE"`
- Untuk mencoba lagi mereplikasi hanya objek yang sebelumnya gagal direplikasi, hanya sertakan. `"FAILED"`
- Untuk mereplikasi objek yang ada dan mencoba lagi mereplikasi objek yang sebelumnya gagal direplikasi, sertakan keduanya dan. `"NONE" "FAILED"`
- Untuk mengisi ulang bucket tujuan dengan objek yang telah direplikasi ke tujuan lain, sertakan. `"COMPLETED"`
- Untuk mereplikasi objek yang sebelumnya direplikasi, sertakan. `"REPLICA"`

Laporan penyelesaian Replikasi Batch

Saat Anda membuat tugas Replikasi Batch, Anda dapat meminta laporan penyelesaian CSV. Laporan ini menunjukkan objek, kode keberhasilan atau kegagalan replikasi, output, dan deskripsi. Untuk informasi selengkapnya tentang laporan pelacakan dan penyelesaian pekerjaan, lihatLaporan penyelesaian.

Untuk daftar kode kegagalan replikasi dan deskripsi, lihat. Penyebab kegagalan replikasi Amazon S3

Untuk informasi tentang pemecahan masalah Replikasi Batch, lihat. Kesalahan Replikasi Batch

Memulai Replikasi Batch

Untuk mempelajari cara menggunakan Replikasi Batch lebih lanjut, lihat Tutorial: Mereplikasi objek yang ada di bucket Amazon S3 Anda dengan Replikasi Batch S3.

Mengonfigurasi kebijakan IAM untuk Replikasi Batch

Karena Replikasi Batch S3 adalah jenis tugas Operasi Batch, Anda harus membuat peran (IAM) AWS Identity and Access Management Operasi Batch untuk memberi Amazon S3 izin untuk melakukan tindakan atas nama Anda. Anda juga harus melampirkan kebijakan IAM Replikasi Batch ke peran IAM Operasi Batch. Contoh berikut membuat peran IAM yang memberi Operasi Batch izin untuk memulai tugas Replikasi Batch.

Membuat peran dan kebijakan IAM

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bawah Manajemen akses, pilih Peran.
3. Pilih Buat peran.
4. Pilih Layanan AWS sebagai jenis entitas tepercaya, Amazon S3 sebagai layanan, dan Operasi Batch S3 sebagai kasus penggunaan.
5. Pilih Berikutnya: Izin.
6. Pilih Buat Kebijakan.
7. Pilih JSON dan masukkan salah satu kebijakan berikut berdasarkan manifes Anda.

> **Note**
>
> Izin yang berbeda diperlukan jika Anda membuat atau memasok manifes. Untuk informasi lebih lanjut lihat,Menentukan manifes untuk tugas Replikasi Batch.

Kebijakan jika menggunakan dan menyimpan manifes yang dihasilkan S3

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Action":[
            "s3:InitiateReplication"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** replication source bucket ***/*"
         ]
      },
      {
         "Action":[
            "s3:GetReplicationConfiguration",
            "s3:PutInventoryConfiguration"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** replication source bucket ***"
         ]
      },
      {
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** manifest bucket ***/*"
         ]
      },
      {
```

```
            "Effect":"Allow",
            "Action":[
               "s3:PutObject"
            ],
            "Resource":[
               "arn:aws:s3:::*** completion report bucket ****/*",
               "arn:aws:s3:::*** manifest bucket ****/*"
            ]
         }
      ]
}
```

Kebijakan jika menggunakan manifes yang disediakan pengguna

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Action":[
            "s3:InitiateReplication"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** replication source bucket ***/*"
         ]
      },
            {
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** manifest bucket ***/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject"
         ],
         "Resource":[
            "arn:aws:s3:::*** completion report bucket ****/*"
```

```
            ]
        }
    ]
}
```

8. Pilih Berikutnya: Tanda.
9. Pilih Berikutnya: Tinjau.
10. Pilih nama untuk kebijakan tersebut dan pilih Buat kebijakan.
11. Lampirkan kebijakan ini ke peran Anda dan pilih Berikutnya: Tanda.
12. Pilih Berikutnya: Tinjau.
13. Pilih nama untuk kebijakan tersebut dan pilih Buat peran.

Verifikasi kebijakan kepercayaan

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di bawah Manajemen akses, pilih Peran, dan pilih peran yang baru dibuat.
3. Pilih Percaya hubungan, Edit hubungan kepercayaan.
4. Pastikan bahwa peran ini menggunakan kebijakan kepercayaan berikut:

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"batchoperations.s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

Membuat tugas Replikasi Batch untuk aturan replikasi pertama atau tujuan baru

Saat Anda membuat aturan pertama dalam konfigurasi replikasi baru atau menambahkan tujuan baru ke konfigurasi yang ada melalui AWS Management Console, Anda dapat membuat pekerjaan Replikasi Batch secara opsional.

Untuk menggunakan Replikasi Batch untuk konfigurasi yang ada tanpa menambahkan tujuan baru lihat, Membuat tugas Replikasi Batch untuk aturan replikasi yang ada.

Menggunakan Batch Replication untuk aturan replikasi baru atau tujuan melalui AWS Management Console

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang berisi objek yang ingin direplikasi.
3. Untuk membuat aturan replikasi baru atau mengedit aturan yang ada, pilih Manajemen, dan gulir ke bawah ke Aturan replikasi:

   - Untuk membuat aturan replikasi baru, pilih Buat aturan replikasi.

     > **Note**
     >
     > Untuk contoh tentang cara menyiapkan aturan replikasi dasar lihat, Contoh untuk mengonfigurasi replikasi langsung.

   - Untuk mengedit aturan replikasi yang ada, pilih aturan, lalu pilih Edit aturan.
4. Buat aturan replikasi baru Anda atau edit tujuan untuk aturan replikasi yang ada, lalu pilih Simpan.

   Setelah membuat aturan pertama dalam konfigurasi replikasi baru atau mengedit konfigurasi yang ada untuk menambahkan tujuan baru, dialog Replikasi objek lama? akan muncul, yang memberi Anda opsi untuk membuat tugas Replikasi Batch.
5. Jika Anda ingin menjalankan tugas ini sekarang, pilih Ya, replikasi objek lama.

   Jika Anda ingin menjalankan tugas ini di lain waktu, pilih Tidak, jangan replikasi objek lama.
6. Buat tugas Replikasi Batch S3 Anda. Tugas Replikasi Batch S3 memiliki beberapa pengaturan:

Opsi menjalankan Tugas

Jika Anda ingin tugas Replikasi Batch S3 langsung berjalan, Anda dapat memilih Jalankan tugas secara otomatis saat siap. Jika Anda ingin menjalankan tugas di lain waktu, pilih Tugas menunggu untuk dijalankan saat siap.

Jika Anda memilih Jalankan tugas secara otomatis saat siap, Anda tidak akan dapat membuat dan menyimpan manifes Operasi Batch. Untuk menyimpan manifes Operasi Batch, pilih Tugas menunggu untuk dijalankan saat siap.

Manifes Operasi Batch

Manifes adalah daftar semua objek berisi tindakan tertentu yang ingin Anda jalankan. Anda dapat memilih untuk menyimpan manifes Operasi Batch. Mirip dengan file Persediaan S3, manifes akan disimpan sebagai file CSV dan disimpan dalam bucket. Untuk mempelajari manifes Operasi Batch lebih lanjut, lihat Menentukan manifes.

Laporan penyelesaian

Operasi Batch S3 mengeksekusi satu tugas untuk setiap objek yang ditentukan dalam manifes. Laporan penyelesaian memberikan cara mudah untuk melihat hasil tugas Anda dalam format terkonsolidasi tanpa perlu pengaturan tambahan. Anda dapat meminta laporan penyelesaian untuk semua tugas atau hanya untuk tugas yang gagal. Untuk mempelajari laporan penyelesaian lebih lanjut, lihat Laporan penyelesaian.

Izin

Salah satu penyebab paling umum kegagalan replikasi adalah izin yang tidak memadai dalam peran yang disediakan AWS Identity and Access Management (IAM). Untuk informasi selengkapnya tentang membuat peran ini, lihat Mengonfigurasi kebijakan IAM untuk Replikasi Batch.

7. Pilih Buat tugas Operasi Batch.

Membuat tugas Replikasi Batch untuk aturan replikasi yang ada

Anda dapat mengonfigurasi Replikasi Batch S3 untuk konfigurasi replikasi yang ada dengan menggunakan AWS SDK, AWS Command Line Interface (AWS CLI), atau konsol Amazon S3. Untuk ikhtisar Replikasi Batch lihat, Mereplikasi objek yang ada dengan Replikasi Batch S3.

Sebagai prasyarat, Anda harus membuat peran Operasi Batch AWS Identity and Access Management (IAM) untuk memberikan izin Amazon S3 untuk melakukan tindakan atas nama Anda, lihat. Mengonfigurasi kebijakan IAM untuk Replikasi Batch

Ketika tugas Replikasi Batch selesai, Anda menerima laporan penyelesaian. Untuk informasi selengkapnya tentang cara menggunakan laporan untuk memeriksa tugas, lihat Melacak status tugas dan laporan penyelesaian.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pilih Operasi Batch di panel navigasi konsol Amazon S3.
3. Pilih Buat tugas.
4. Pilih Wilayah tempat Anda ingin membuat tugas Anda.
5. Pilih format Manifes. Contoh ini akan menunjukkan cara membuat manifes berdasarkan konfigurasi replikasi S3 yang ada.

> **Note**
>
> Manifes adalah daftar semua objek berisi tindakan tertentu yang ingin Anda jalankan. Untuk mempelajari manifes Operasi Batch lebih lanjut, lihat Menentukan manifes. Jika Anda memiliki manifes yang sudah siap, pilih laporan persediaan S3 (manifest.json) atau CSV. Jika objek dalam manifes Anda berada dalam bucket berversi, Anda harus menentukan ID versi untuk objek tersebut. Untuk informasi selengkapnya tentang cara membuat manifes, lihat Menentukan manifes.

6. Untuk membuat manifes berdasarkan konfigurasi replikasi Anda, pilih Buat manifes menggunakan konfigurasi Replikasi S3. Kemudian pilih bucket sumber konfigurasi replikasi Anda.
7. (Opsional) Anda dapat menyertakan filter tambahan seperti tanggal pembuatan objek dan status replikasi. Untuk contoh tentang cara memfilter berdasarkan status replikasi lihat, Menentukan manifes untuk tugas Replikasi Batch.
8. Untuk menyimpan manifes, pilih Simpan manifes Operasi Batch.
   a. Jika Anda memilih untuk membuat dan menyimpan manifes, Anda harus memilih Bucket di akun ini atau Bucket di Akun AWS lain. Tentukan nama bucket di kotak teks.

> **Note**
>
> Manifes yang dihasilkan harus disimpan Wilayah AWS sama dengan bucket sumber.

b. Pilih Tipe enkripsi.

9. (Opsional) Berikan Deskripsi.
10. Sesuaikan Prioritas tugas jika diperlukan. Angka yang lebih tinggi menunjukkan prioritas yang lebih tinggi. Amazon S3 mencoba menjalankan tugas dengan prioritas lebih tinggi sebelum tugas dengan prioritas lebih rendah. Untuk informasi selengkapnya tentang prioritas tugas, lihat Menetapkan prioritas tugas.
11. (Opsional) Buat laporan penyelesaian. Untuk membuatnya, pilih Buat laporan penyelesaian.

    Jika Anda memilih untuk membuat laporan penyelesaian, Anda harus memilih untuk melaporkan Hanya tugas gagal atau Semua tugas, dan menyediakan bucket tujuan untuk laporan tersebut.
12. Pilih peran IAM yang valid.

> **Note**
>
> Untuk informasi selengkapnya tentang cara membuat peran IAM, lihat Mengonfigurasi kebijakan IAM untuk Replikasi Batch.

13. (Opsional) Tambahkan tag pekerjaan ke tugas Replikasi Batch.
14. Pilih Selanjutnya.
15. Tinjau konfigurasi tugas Anda dan pilih Buat tujgas.

Menggunakan AWS CLI manifes dengan S3

Contoh berikut membuat pekerjaan Replikasi Batch S3 menggunakan manifes yang dihasilkan S3 untuk. Akun AWS *111122223333* Contoh ini akan mencoba mereplikasi objek lama dan objek yang sebelumnya gagal direplikasi. Untuk informasi tentang pemfilteran berdasarkan status replikasi lihat, Menentukan manifes untuk tugas Replikasi Batch.

```
aws s3control create-job --account-id 111122223333 --operation
 '{"S3ReplicateObject":{}}' --report '{"Bucket":"arn:aws:s3:::***
 completion report bucket ****","Prefix":"batch-replication-report",
```

```
 "Format":"Report_CSV_20180820","Enabled":true,"ReportScope":"AllTasks"}'
 --manifest-generator '{"S3JobManifestGenerator": {"ExpectedBucketOwner":
 "111122223333", "SourceBucket": "arn:aws:s3:::*** replication source bucket
 ***", "EnableManifestOutput": false, "Filter": {"EligibleForReplication": true,
 "ObjectReplicationStatuses": ["NONE","FAILED"]}}}' --priority 1 --role-arn
 arn:aws:iam::111122223333:role/batch-Replication-IAM-policy --no-confirmation-required
 --region source-bucket-region
```

> **Note**
>
> Pekerjaan harus dimulai dari bucket sumber Wilayah AWS replikasi yang sama. Peran IAM `role/batch-Replication-IAM-policy` telah dibuat sebelumnyaj. Lihat Mengonfigurasi kebijakan IAM untuk Replikasi Batch.

Setelah berhasil memulai tugas Replikasi Batch, Anda menerima ID tugas sebagai respons. Anda dapat memantau tugas jini menggunakan perintah berikut.

```
aws s3control describe-job --account-id 111122223333 --job-id job-id --region source-
bucket-region
```

Menggunakan AWS CLI dengan manifes yang disediakan pengguna

Contoh berikut membuat tugas Replikasi Batch S3 menggunakan manifes yang ditentukan pengguna untuk Akun AWS *111122223333*. Jika objek dalam manifes Anda berada dalam bucket berversi, Anda harus menentukan ID versi untuk objek tersebut. Hanya objek dengan ID versi yang ditentukan dalam manifes yang akan direplikasi. Untuk informasi selengkapnya tentang cara membuat manifes, lihat Menentukan manifes.

```
aws s3control create-job --account-id 111122223333 --operation
 '{"S3ReplicateObject":{}}' --report '{"Bucket":"arn:aws:s3:::***
 completion report bucket ****","Prefix":"batch-replication-report",
 "Format":"Report_CSV_20180820","Enabled":true,"ReportScope":"AllTasks"}'
 --manifest '{"Spec":{"Format":"S3BatchOperations_CSV_20180820","Fields":
["Bucket","Key","VersionId"]},"Location":{"ObjectArn":"arn:aws:s3:::*** completion
 report bucket ****/manifest.csv","ETag":"Manifest Etag"}}' --priority 1 --role-arn
 arn:aws:iam::111122223333:role/batch-Replication-IAM-policy --no-confirmation-required
 --region source-bucket-region
```

> **Note**
>
> Pekerjaan harus dimulai dari bucket sumber Wilayah AWS replikasi yang sama. Peran IAM `role/batch-Replication-IAM-policy` telah dibuat sebelumnyaj. Lihat Mengonfigurasi kebijakan IAM untuk Replikasi Batch.

Setelah berhasil memulai tugas Replikasi Batch, Anda menerima ID tugas sebagai respons. Anda dapat memantau tugas jini menggunakan perintah berikut.

```
aws s3control describe-job --account-id 111122223333 --job-id job-id --region source-
bucket-region
```

Mengategorikan penyimpanan Anda menggunakan tag

Gunakan pemberian tag objek untuk mengategorikan penyimpanan. Setiap tag adalah pasangan nilai kunci.

Anda dapat menambahkan tag ke objek baru saat mengunggahnya, atau Anda dapat menambahkannya ke objek yang sudah ada.

- Anda dapat mengaitkan hingga 10 tag dengan objek. Tag yang terkait dengan sebuah objek harus memiliki kunci tag unik.
- Kunci tanda dapat terdiri dari hingga 128 karakter Unicode, dan nilai tanda dapat terdiri dari hingga 256 karakter Unicode. Tag objek Amazon S3 diwakili secara internal dalam UTF-16. Perhatikan bahwa dalam UTF-16, karakter menggunakan 1 atau 2 posisi karakter.
- Kunci dan nilai peka terhadap huruf besar dan kecil.
- Untuk informasi selengkapnya tentang pembatasan tag, lihat Pembatasan tag yang ditentukan pengguna di Panduan Pengguna AWS Billing and Cost Management. Untuk pembatasan tag dasar, lihat Pembatasan tag di Panduan Pengguna Amazon EC2.

Contoh

Pertimbangkan contoh pemberian tag berikut:

Example Informasi PHI

Misalkan objek berisi data informasi kesehatan yang dilindungi (PHI). Anda dapat memberi tag pada objek menggunakan pasangan nilai kunci berikut.

```
PHI=True
```

atau

```
Classification=PHI
```

Example File proyek

Misalkan, Anda menyimpan berkas proyek dalam bucket S3. Anda dapat memberi tag objek ini dengan kunci yang diberi nama `Project` dan nilai, seperti yang ditunjukkan berikut ini.

```
Project=Blue
```

Example Banyak tag

Anda dapat menambahkan beberapa tag ke sebuah objek, seperti yang ditunjukkan berikut.

```
Project=x
Classification=confidential
```

Prefiks nama kunci dan tag

Prefiks kunci objek juga memungkinkan Anda untuk mengategorikan penyimpanan. Namun, kategorisasi berbasis prefiks bersifat satu dimensi. Pertimbangkan nama kunci objek berikut:

```
photos/photo1.jpg
project/projectx/document.pdf
project/projecty/document2.pdf
```

Nama-nama kunci ini memiliki prefiks `photos/`, `project/projectx/`, dan `project/projecty/`. Prefiks ini mengaktifkan kategorisasi satu dimensi. Artinya, semua yang berada di bawah prefix adalah satu kategori. Misalnya, prefiks `project/projectx` mengidentifikasi semua dokumen yang terkait dengan proyek x.

Dengan pemberian tag, Anda sekarang memiliki dimensi lain. Jika Anda ingin foto1 dalam kategori proyek x, Anda dapat memberi tag objek yang sesuai.

Manfaat tambahan

Selain klasifikasi data, pemberian tag menawarkan manfaat seperti berikut ini:

- Tag objek memungkinkan kontrol izin akses yang detail. Misalnya, Anda dapat memberikan izin pengguna ke objek hanya-baca dengan tag tertentu.
- Tag objek memungkinkan manajemen siklus hidup objek yang detail yaitu Anda dapat menentukan filter berbasis tag, selain prefiks nama kunci, dalam aturan siklus hidup.
- Saat menggunakan analitik Amazon S3, Anda dapat mengonfigurasi filter agar membuat grup objek secara bersama-sama untuk analisis dengan tag objek, dengan prefiks nama kunci, atau dengan prefiks dan tag.
- Anda juga dapat menyesuaikan CloudWatch metrik Amazon untuk menampilkan informasi berdasarkan filter tag tertentu. Bagian berikut memberikan perincian.

> **Important**
>
> Dapat menggunakan tag untuk melabeli objek yang berisi data rahasia, seperti informasi pengenal pribadi (PII) atau informasi kesehatan terlindungi (PHI). Namun demikian, tag itu sendiri seharusnya tidak berisi informasi rahasia.

Menambahkan set tag objek ke beberapa objek Amazon S3 dengan permintaan tunggal

Untuk menambahkan set tag objek ke lebih dari satu objek Amazon S3 dengan permintaan tunggal, Anda dapat menggunakan Operasi Batch S3. Anda menyediakan daftar objek yang akan dioperasikan kepada Operasi Batch S3. Operasi Batch S3 akan memanggil masing-masing operasi API untuk melakukan operasi tertentu. Satu tugas Operasi Batch dapat melakukan operasi tertentu pada miliaran objek yang berisi data sebesar eksabita.

Operasi Batch S3 memiliki fitur melacak progres, mengirimkan notifikasi, dan menyimpan laporan penyelesaian terperinci dari semua tindakan, menyediakan pengalaman yang dikelola sepenuhnya, dapat diaudit, dan nirserver. Anda dapat menggunakan Operasi Batch S3 melalui konsol Amazon S3, SDK AWS CLI AWS , atau REST API. Untuk informasi selengkapnya, lihat the section called “Dasar-dasar Operasi Batch”.

Untuk informasi selengkapnya tentang tag objek, lihat Mengelola tag objek.

Operasi API terkait pemberian tag objek

Amazon S3 mendukung operasi API berikut yang khusus untuk pemberian tag objek:

Operasi API objek

- Pemberian tag PUT Objek–Mengganti tag pada satu objek. Anda menentukan tag di isi permintaan. Ada dua skenario berbeda dalam pengelolaan tag objek menggunakan API ini.
  - Objek tidak memiliki tag–Dengan menggunakan API ini, Anda dapat menambahkan satu set tag ke objek (objek tidak memiliki tag sebelumnya).
  - Objek memiliki satu set tag yang sudah ada–Untuk memodifikasi tag yang sudah ada, Anda harus mengambil set tag yang sudah ada terlebih dahulu, mengubahnya dari sisi klien, kemudian menggunakan API ini untuk mengganti set tag tersebut.

> **Note**
>
> Jika Anda mengirim permintaan ini dengan set tag kosong, Amazon S3 menghapus tag yang ada yang ditetapkan di objek. Jika Anda menggunakan metode ini, Anda akan dikenakan biaya untuk Permintaan Tingkat 1 (PUT). Untuk informasi selengkapnya, lihat harga Amazon S3.
> Permintaan pemberian tag DELETE Object lebih disarankan karena mencapai hasil yang sama tanpa dikenakan biaya.

- Pemberian tag GET Object–Mengembalikan set tag yang terkait dengan objek. Amazon S3 mengembalikan tag objek di dalam isi respons.
- Pemberian tag DELETE Object–Menghapus set tag yang terkait dengan objek.

Operasi API lainnya yang mendukung pemberian tag

- PUT Objek dan Memulai Pengunggahan Multibagian– Anda dapat menentukan tag saat membuat objek. Anda menentukan tag menggunakan header permintaan `x-amz-tagging`.
- GET Object–Alih-alih mengembalikan set tag, Amazon S3 mengembalikan jumlah tag objek dalam header `x-amz-tag-count` (hanya jika pemohon memiliki izin untuk membaca tag) karena ukuran respons header terbatas hingga 8 K byte. Jika Anda ingin melihat tag, Anda membuat permintaan lain untuk operasi API pemberian tag GET Object.

- Objek POST–Anda dapat menentukan tag di permintaan POST.

  Selama tag dalam permintaan Anda tidak melebihi batas ukuran header permintaan HTTP 8 K byte, Anda dapat menggunakan API `PUT Object` untuk membuat objek dengan tag. Jika tag yang Anda tetapkan melebihi batas ukuran header, Anda dapat menggunakan metode POST ini yaitu Anda menyertakan tag di dalam bodi.

  PUT Objek-Salin–Anda dapat menentukan `x-amz-tagging-directive` dalam permintaan Anda untuk mengarahkan Amazon S3 agar menyalin (perilaku default) tag atau mengganti tag dengan set tag baru yang diberikan dalam permintaan.

Perhatikan hal berikut:

- Pemberian Tag Objek S3 sangat konsisten. Untuk informasi selengkapnya, lihat Model konsistensi data Amazon S3.

Konfigurasi tambahan

Bagian ini menjelaskan bagaimana pemberian tag objek berkaitan dengan konfigurasi lain.

Pemberian tag objek dan manajemen siklus hidup

Dalam konfigurasi siklus operasional bucket, Anda dapat menetapkan filter untuk memilih subset objek yang aturannya berlaku. Anda dapat menentukan filter berdasarkan prefiks nama kunci, tag objek, atau keduanya.

Misalkan Anda menyimpan foto (format mentah dan yang sudah jadi) dalam bucket Amazon S3. Anda dapat memberi tag objek ini seperti yang ditunjukkan berikut ini.

```
phototype=raw
or
phototype=finished
```

Anda dapat mempertimbangkan untuk mengarsipkan foto mentah ke S3 Glacier beberapa saat setelah foto tersebut dibuat. Anda dapat mengonfigurasi aturan siklus aktif dengan filter yang mengidentifikasi subset objek dengan prefiks nama kunci (`photos/`) yang memiliki tag tertentu (`phototype=raw`).

Untuk informasi selengkapnya, lihat Mengelola siklus hidup penyimpanan Anda.

Pemberian tag dan replikasi objek

Jika Anda mengonfigurasi Replikasi pada bucket, Amazon S3 mereplikasi tag, asalkan Anda memberikan izin Amazon S3 untuk membaca tag tersebut. Untuk informasi selengkapnya, lihat Menyiapkan replikasi langsung.

Pemberitahuan peristiwa pemberian tag objek

Anda dapat mengatur pemberitahuan peristiwa Amazon S3 untuk menerima pemberitahuan saat tag objek ditambahkan atau dihapus dari objek. Jenis peristiwa `s3:ObjectTagging:Put` memberi tahu Anda saat tag PUT pada objek atau saat tag yang ada diperbarui. Jenis peristiwa `s3:ObjectTagging:Delete` memberi tahu Anda saat tag dihapus dari objek. Untuk informasi selengkapnya, lihat Mengaktifkan pemberitahuan peristiwa.

Untuk informasi lebih lanjut tentang pemberian tag objek, lihat topik berikut ini:

Topik

- Kebijakan pemberian tag dan kontrol akses
- Mengelola tag objek

Kebijakan pemberian tag dan kontrol akses

Anda juga dapat menggunakan kebijakan izin (bucket dan kebijakan pengguna) untuk mengelola izin terkait pemberian tag objek. Untuk tindakan kebijakan, lihat topik berikut:

- Operasi objek
- Operasi bucket

Tag objek memungkinkan kontrol akses yang detail untuk mengelola izin. Anda dapat memberikan izin bersyarat berdasarkan tag objek. Amazon S3 mendukung kunci kondisi berikut yang dapat Anda gunakan untuk memberikan izin bersyarat berdasarkan tag objek:

- `s3:ExistingObjectTag/<tag-key>`–Gunakan kunci kondisi ini untuk memverifikasi bahwa tag objek yang ada memiliki kunci dan nilai tag spesifik.

> **Note**
>
> Saat memberikan izin untuk operasi `PUT Object` dan `DELETE Object`, kunci kondisi ini tidak didukung. Artinya, Anda tidak dapat membuat kebijakan memberikan atau menolak izin pengguna untuk menghapus atau menimpa objek berdasarkan tag yang ada.

- `s3:RequestObjectTagKeys`–Gunakan kunci kondisi ini untuk membatasi kunci tag yang ingin Anda izinkan pada objek. Ini berguna saat menambahkan tag ke objek menggunakan permintaan objek PutObjectTagging dan PutObject, dan POST.
- `s3:RequestObjectTag/<tag-key>`–Gunakan kunci kondisi ini untuk membatasi kunci dan nilai tag yang ingin Anda izinkan pada objek. Ini berguna saat menambahkan tag ke objek menggunakan permintaan PutObjectTagging dan PutObject, dan POST Bucket.

Untuk daftar lengkap kunci kondisi Amazon S3, lihat Contoh kebijakan bucket menggunakan tombol kondisi. Kebijakan izin berikut menggambarkan cara pemberian tag objek yang memungkinkan manajemen izin akses yang detail.

Example 1: Mengizinkan pengguna untuk membaca hanya objek yang memiliki tag dan nilai kunci tertentu

Kebijakan izin berikut membatasi pengguna agar hanya membaca objek yang memiliki kunci dan nilai tag `environment: production`. Kebijakan ini menggunakan kunci kondisi `s3:ExistingObjectTag` untuk menentukan kunci tag dan nilai.

```
{
  "Version": "2012-10-17",
  "Statement": [
  {
    "Principal": {
      "AWS": [
        "arn:aws:iam::111122223333:role/JohnDoe"
      ]
    },
    "Effect": "Allow",
    "Action": ["s3:GetObject", "s3:GetObjectVersion"],
    "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
    "Condition": {
      "StringEquals":
        {"s3:ExistingObjectTag/environment": "production"}
```

```
    }
  }
  ]
}
```

Example 2: Batasi kunci tag objek mana yang dapat ditambahkan pengguna

Kebijakan izin berikut memberikan izin pengguna untuk melakukan tindakan `s3:PutObjectTagging`, yang memungkinkan pengguna menambahkan tag ke objek yang sudah ada. Kondisi ini menggunakan kunci kondisi `s3:RequestObjectTagKeys` untuk menentukan kunci tag yang diizinkan, seperti `Owner` atau `CreationDate`. Untuk informasi selengkapnya, lihat Membuat kondisi yang menguji beberapa nilai kunci dalam Panduan Pengguna IAM.

Kebijakan ini memastikan agar setiap kunci tag yang ditentukan dalam permintaan adalah kunci tag yang diotorisasi. Kualifikasi `ForAnyValue` dalam kondisi tersebut memastikan bahwa setidaknya ada satu kunci yang ditentukan dalam permintaan.

```
{
   "Version": "2012-10-17",
  "Statement": [
    {"Principal":{"AWS":[
            "arn:aws:iam::111122223333:role/JohnDoe"
         ]
       },
 "Effect": "Allow",
      "Action": [
        "s3:PutObjectTagging"
      ],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {"ForAnyValue:StringEquals": {"s3:RequestObjectTagKeys": [
            "Owner",
            "CreationDate"
          ]
        }
      }
    }
  ]
}
```

Example 3: Memerlukan kunci dan nilai tag spesifik saat mengizinkan pengguna untuk menambahkan tag objek

Kebijakan contoh berikut memberikan izin pengguna untuk melakukan `s3:PutObjectTagging` tindakan, yang memungkinkan pengguna menambahkan tag ke objek yang sudah ada. Kondisi ini mengharuskan pengguna untuk menyertakan kunci tag tertentu (seperti *`Project`*) dengan set nilai ke *`X`*.

```
{
   "Version": "2012-10-17",
  "Statement": [
    {"Principal":{"AWS":[
       "arn:aws:iam::111122223333:user/JohnDoe"
         ]
       },
      "Effect": "Allow",
      "Action": [
        "s3:PutObjectTagging"
      ],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {"StringEquals": {"s3:RequestObjectTag/Project": "X"
        }
      }
    }
  ]
}
```

Mengelola tag objek

Bagian ini menjelaskan bagaimana Anda dapat mengelola tag objek menggunakan AWS SDK untuk Java dan .NET atau konsol Amazon S3.

Penandaan objek memberi Anda cara untuk mengategorikan penyimpanan. Setiap tag adalah pasangan nilai kunci yang mematuhi aturan berikut:

- Anda dapat mengaitkan hingga 10 tag dengan objek. Tag yang terkait dengan sebuah objek harus memiliki kunci tag unik.

- Kunci tanda dapat terdiri dari hingga 128 karakter Unicode, dan nilai tanda dapat terdiri dari hingga 256 karakter Unicode. Tag objek Amazon S3 diwakili secara internal dalam UTF-16. Perhatikan bahwa dalam UTF-16, karakter menggunakan 1 atau 2 posisi karakter.
- Kunci dan nilai peka terhadap huruf besar dan kecil.

Untuk informasi selengkapnya tentang tag objek, lihat Mengategorikan penyimpanan Anda menggunakan tag. Untuk informasi selengkapnya tentang pembatasan tag, lihat Pembatasan Tag yang Ditentukan Pengguna dalam AWS Billing and Cost Management Panduan Pengguna.

Menggunakan konsol S3

Untuk menambahkan tag ke objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang berisi objek yang ingin Anda tambahkan tag.

   Anda juga dapat pergi ke folder.
3. Pada daftar Objek, pilih kotak centang di samping nama objek yang ingin Anda tambahkan tag.
4. Di menu Tindakan, pilih Edit tag.
5. Tinjau objek yang tercantum, lalu pilih Tambahkan tag.
6. Setiap tag objek adalah pasangan nilai kunci. Masukkan Kunci dan Nilai. Untuk menambahkan tag lainnya, pilih Tambahkan tag.

   Anda dapat memasukkan hingga 10 tag untuk objek.
7. Pilih Simpan perubahan.

   Amazon S3 menambahkan tag ke objek tertentu.

Untuk informasi selengkapnya, lihat juga Melihat properti objek di konsol Amazon S3 dan Mengunggah Objek dalam panduan ini.

Menggunakan AWS SDK

Java

Contoh berikut menunjukkan bagaimana menggunakan untuk mengatur tag untuk objek baru dan mengambil atau mengganti tag untuk objek yang ada. AWS SDK for Java Untuk

informasi selengkapnya tentang pemberian tag objek, lihat Mengategorikan penyimpanan Anda menggunakan tag. Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.File;
import java.util.ArrayList;
import java.util.List;

public class ManagingObjectTags {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";
        String keyName = "*** Object key ***";
        String filePath = "*** File path ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Create an object, add two new tags, and upload the object to Amazon
 S3.
            PutObjectRequest putRequest = new PutObjectRequest(bucketName, keyName,
 new File(filePath));
            List<Tag> tags = new ArrayList<Tag>();
            tags.add(new Tag("Tag 1", "This is tag 1"));
            tags.add(new Tag("Tag 2", "This is tag 2"));
            putRequest.setTagging(new ObjectTagging(tags));
            PutObjectResult putResult = s3Client.putObject(putRequest);

            // Retrieve the object's tags.
```

```
            GetObjectTaggingRequest getTaggingRequest = new
 GetObjectTaggingRequest(bucketName, keyName);
            GetObjectTaggingResult getTagsResult =
 s3Client.getObjectTagging(getTaggingRequest);

            // Replace the object's tags with two new tags.
            List<Tag> newTags = new ArrayList<Tag>();
            newTags.add(new Tag("Tag 3", "This is tag 3"));
            newTags.add(new Tag("Tag 4", "This is tag 4"));
            s3Client.setObjectTagging(new SetObjectTaggingRequest(bucketName,
 keyName, new ObjectTagging(newTags)));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Contoh berikut menunjukkan bagaimana menggunakan AWS SDK for .NET untuk mengatur tag untuk objek baru dan mengambil atau mengganti tag untuk objek yang ada. Untuk informasi selengkapnya tentang pemberian tag objek, lihat Mengategorikan penyimpanan Anda menggunakan tag.

Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
```

```
    public class ObjectTagsTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string keyName = "*** key name for the new object ***";
        private const string filePath = @"*** file path ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            PutObjectWithTagsTestAsync().Wait();
        }

        static async Task PutObjectWithTagsTestAsync()
        {
            try
            {
                // 1. Put an object with tags.
                var putRequest = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    FilePath = filePath,
                    TagSet = new List<Tag>{
                        new Tag { Key = "Keyx1", Value = "Value1"},
                        new Tag { Key = "Keyx2", Value = "Value2" }
                    }
                };

                PutObjectResponse response = await
client.PutObjectAsync(putRequest);
                // 2. Retrieve the object's tags.
                GetObjectTaggingRequest getTagsRequest = new GetObjectTaggingRequest
                {
                    BucketName = bucketName,
                    Key = keyName
                };

                GetObjectTaggingResponse objectTags = await
client.GetObjectTaggingAsync(getTagsRequest);
                for (int i = 0; i < objectTags.Tagging.Count; i++)
```

```
                    Console.WriteLine("Key: {0}, Value: {1}",
objectTags.Tagging[i].Key, objectTags.Tagging[i].Value);


                // 3. Replace the tagset.

                Tagging newTagSet = new Tagging();
                newTagSet.TagSet = new List<Tag>{
                    new Tag { Key = "Key3", Value = "Value3"},
                    new Tag { Key = "Key4", Value = "Value4" }
                };


                PutObjectTaggingRequest putObjTagsRequest = new
PutObjectTaggingRequest()
                {
                    BucketName = bucketName,
                    Key = keyName,
                    Tagging = newTagSet
                };
                PutObjectTaggingResponse response2 = await
client.PutObjectTaggingAsync(putObjTagsRequest);

                // 4. Retrieve the object's tags.
                GetObjectTaggingRequest getTagsRequest2 = new
GetObjectTaggingRequest();
                getTagsRequest2.BucketName = bucketName;
                getTagsRequest2.Key = keyName;
                GetObjectTaggingResponse objectTags2 = await
client.GetObjectTaggingAsync(getTagsRequest2);
                for (int i = 0; i < objectTags2.Tagging.Count; i++)
                    Console.WriteLine("Key: {0}, Value: {1}",
objectTags2.Tagging[i].Key, objectTags2.Tagging[i].Value);

            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine(
                        "Error encountered ***. Message:'{0}' when writing an
object"
                        , e.Message);
            }
            catch (Exception e)
            {
```

```
                Console.WriteLine(
                    "Encountered an error. Message:'{0}' when writing an object"
                    , e.Message);
            }
        }
    }
}
```

Menggunakan tag alokasi biaya bucket S3

Untuk melacak biaya penyimpanan atau kriteria lain untuk proyek individu atau kelompok proyek, beri label pada bucket Amazon S3 Anda dengan menggunakan tag alokasi biaya. Tag alokasi biaya adalah pasangan nilai kunci yang Anda kaitkan dengan bucket S3. Setelah Anda mengaktifkan tanda alokasi biaya, AWS menggunakan tanda untuk mengatur biaya sumber daya pada laporan alokasi biaya Anda. Tag alokasi biaya hanya dapat digunakan untuk melabeli bucket. Untuk informasi tentang tag yang digunakan untuk melabeli objek, lihat Mengategorikan penyimpanan Anda menggunakan tag.

Laporan alokasi biaya mencantumkanAWS penggunaan akun Anda menurut kategori produk dan pengguna akun tertaut. Laporan berisi item garis yang sama dengan laporan penagihan terperinci (lihat Memahami laporan AWS penagihan dan penggunaan Anda untuk Amazon S3) dan kolom tambahan untuk kunci tag Anda.

AWSmenyediakan dua jenis tag alokasi biaya, tagAWS -tanda yang dihasilkan dan tag yang ditentukan pengguna. AWSmendefinisikan, membuat, dan menerapkan`createdBy` tagAWS yang dihasilkan untuk Anda setelah CreateBucket peristiwa Amazon S3. Anda menentukan, membuat, dan menerapkan tag yang ditentukan pengguna ke bucket S3.

Anda harus mengaktifkan kedua jenis tag secara terpisah di konsol Manajemen Penagihan dan Biaya sebelum tag tersebut muncul di laporan penagihan Anda. Untuk informasi lebih lanjut tentangAWS -tanda yang dihasilkan, lihat AWS-Tanda Alokasi Biaya yang Dihasilkan.

- Untuk membuat tanda di konsol, lihat Melihat properti untuk bucket S3.
- Untuk membuat tag menggunakan Amazon S3 API, lihat MEMASUKKAN tagging Bucket (PUT Bucket tagging) dalam Referensi Amazon Simple Storage Service API.
- Untuk membuat tag menggunakanAWS CLI, lihat put-bucket-taggingdiAWS CLI Command Reference.

- Untuk informasi lebih lanjut tentang mengaktifkan tanda, lihat Menggunakan tanda alokasi biaya di AWS Billing Panduan Pengguna.

Tag alokasi biaya yang ditentukan pengguna

Tag alokasi biaya yang ditentukan pengguna memiliki komponen berikut:

- Kunci tag. Kunci tag adalah nama tag. Misalnya, pada proyek/Trinity tag, proyek adalah kuncinya. Kunci tag adalah string yang peka huruf besar/kecil yang dapat berisi 1 hingga 128 karakter Unicode.

- Nilai tag. Nilai tag adalah string yang diperlukan. Misalnya, dalam proyek/Trinity tag, Trinity adalah nilai. Nilai tag adalah string yang peka huruf besar/kecil yang dapat berisi 0 hingga 256 karakter Unicode.

Untuk detail tentang karakter yang diizinkan untuk tanda yang ditentukan pengguna dan batasan lainnya, lihat Pembatasan Tanda yang Ditentukan Pengguna di AWS Billing Panduan Pengguna. Untuk informasi lebih lanjut tentang tanda yang ditentukan pengguna, lihat Tanda Alokasi Biaya yang Ditentukan Pengguna di AWS Billing Panduan Pengguna.

Tag bucket S3

Setiap bucket S3 memiliki rangkaian tag. Rangkaian tag berisi semua tag yang ditetapkan pada bucket tersebut. Rangkaian tag dapat berisi 50 tag atau kosong. Kunci dalam rangkaian tag harus unik, tetapi nilai dalam rangkaian tag tidak harus unik. Misalnya, Anda dapat memiliki nilai yang sama dalam rangkaian tag yang disebut proyek/Trinity dan pusat biaya/Trinity.

Dalam bucket, jika Anda menambahkan tag yang memiliki kunci yang sama dengan tag yang ada, nilai yang baru akan menimpa nilai yang lama.

AWS tidak menerapkan makna semantik pada tanda Anda. Kami menafsirkan tag dengan ketat sebagai string karakter.

Untuk menambahkan, mencantumkan, mengedit, atau menghapus tanda, Anda dapat menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), atau API Amazon S3.

Info Selengkapnya

- Menggunakan Tanda Alokasi Biaya di AWS Billing Panduan Pengguna

- Memahami laporan AWS penagihan dan penggunaan Anda untuk Amazon S3
- AWS Billing laporan untuk Amazon S3

Pelaporan penagihan dan penggunaan untuk Amazon S3

> **Important**
>
> Pada 13 Mei 2024, kami mulai menerapkan perubahan untuk menghilangkan biaya atas permintaan tidak sah yang tidak diprakarsai oleh pemilik bucket. Setelah penerapan perubahan ini selesai, pemilik bucket tidak akan pernah dikenakan biaya permintaan atau bandwidth untuk permintaan yang mengembalikan kesalahan `AccessDenied` (HTTP`403 Forbidden`) saat permintaan ini dimulai dari luar AWS akun atau organisasi masing-masing. AWS Untuk informasi selengkapnya tentang daftar lengkap HTTP `3XX` dan kode `4XX` status yang tidak akan ditagih, lihatPenagihan untuk respons kesalahan Amazon S3. Perubahan penagihan ini tidak memerlukan pembaruan untuk aplikasi Anda dan berlaku untuk semua bucket S3. Ketika penerapan perubahan ini selesai di semua Wilayah AWS, kami akan memperbarui dokumentasi kami.

Saat menggunakan Amazon S3, Anda tidak perlu membayar biaya di muka atau berkomitmen untuk berapa banyak konten yang akan Anda simpan. Seperti yang lain Layanan AWS, Anda membayar saat Anda pergi dan hanya membayar untuk apa yang Anda gunakan.

AWS memberikan laporan berikut untuk Amazon S3:

- Laporan penagihan — Beberapa laporan yang memberikan tampilan tingkat tinggi dari semua aktivitas Layanan AWS yang Anda gunakan, termasuk Amazon S3. AWS selalu menagih pemilik ember S3 untuk biaya Amazon S3, kecuali ember dibuat sebagai ember Pemohon Pays. Untuk informasi selengkapnya tentang Pembayaran Pemohon, lihat Menggunakan bucket Pembayaran Pemohon untuk transfer dan penggunaan penyimpanan. Untuk informasi selengkapnya tentang laporan penagihan, lihat AWS Billing laporan untuk Amazon S3.
- Laporan penggunaan–Ringkasan aktivitas untuk layanan tertentu, yang dikumpulkan berdasarkan jam, hari, atau bulan. Anda dapat memilih jenis penggunaan dan pengoperasian yang akan disertakan. Anda juga dapat memilih cara pengumpulan data. Untuk informasi selengkapnya, lihat AWS laporan penggunaan untuk Amazon S3.

Topik berikut menyediakan informasi tentang pelaporan penagihan dan penggunaan untuk Amazon S3.

Topik

- AWS Billing laporan untuk Amazon S3
- AWS laporan penggunaan untuk Amazon S3
- Memahami laporan AWS penagihan dan penggunaan Anda untuk Amazon S3
- Penagihan untuk respons kesalahan Amazon S3

AWS Billing laporan untuk Amazon S3

Tagihan bulanan Anda dari AWS memisahkan informasi penggunaan dan biaya Anda berdasarkan Layanan AWS dan fungsinya. Ada beberapa AWS Billing laporan yang tersedia: laporan bulanan, laporan alokasi biaya, dan laporan penagihan terperinci. Untuk informasi tentang cara melihat laporan penagihan, lihat Melihat Tagihan Anda di Panduan Pengguna AWS Billing .

Untuk melacak AWS penggunaan Anda dan memberikan perkiraan biaya yang terkait dengan akun Anda, Anda dapat mengaturnya AWS Cost and Usage Reports. Untuk informasi lebih lanjut, lihat Apa itu AWSCost and Usage Reports? dalam Panduan Ekspor AWS Data.

Anda juga dapat mengunduh laporan penggunaan yang akan memberikan informasi yang lebih lengkap tentang penggunaan penyimpanan Amazon S3 dibandingkan laporan penagihan. Untuk informasi selengkapnya, lihat AWS laporan penggunaan untuk Amazon S3.

Tabel berikut mencantumkan biaya yang terkait dengan penggunaan Amazon S3.

Biaya penggunaan Amazon S3

Biaya	Komentar
Penyimpanan	Anda membayar untuk penyimpanan objek dalam bucket S3. Tarif yang dikenakan biaya tergantung pada ukuran objek Anda, berapa lama Anda menyimpan objek selama sebulan, dan kelas penyimpanan. Amazon S3 menawarkan kelas penyimpanan berikut: Standar S3, S3 Express One Zone, S3 Intelligent-Tiering, S3 Standard-IA (IA untuk akses

Biaya	Komentar
	yang jarang), S3 One Zone-IA, S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval , S3 Glacier Deep Archive, atau Reduced Redundancy Storage (RRS). Untuk informasi selengkapnya tentang kelas penyimpanan, lihat Menggunakan kelas penyimpanan Amazon S3. Ketahuilah bahwa jika Anda mengaktifkan Versi S3, Anda dikenakan biaya untuk setiap versi objek yang dipertahankan. Untuk informasi selengkapnya tentang penentuan versi, lihat Cara kerja Penentuan Versi S3.
Pemantauan dan otomatisasi	Anda membayar biaya pemantauan dan otomatisasi bulanan per objek yang disimpan di kelas penyimpanan S3 Intelligent-Tiering untuk memantau pola akses dan memindahkan objek di antara tingkat akses di S3 Intelligent-Tiering.
Permintaan	Anda membayar permintaan, misalnya, `GET` permintaan, yang dibuat terhadap ember dan objek S3 Anda. Ini termasuk permintaan siklus hidup. Tarif untuk permintaan tergantung pada jenis permintaan yang Anda buat. Untuk informasi tentang harga permintaan, lihat Harga Amazon S3.
Pengambilan	Anda membayar untuk mengambil objek yang disimpan dalam penyimpanan S3 Standard-IA, S3 One Zone-IA, S3 Glacier Instant, S3 Glacier Flexible Retrieval, dan penyimpanan S3 Glacier Deep Archive.

Biaya	Komentar
Menghapus lebih awal	Jika Anda menghapus objek yang disimpan di penyimpanan S3 Standard-IA, S3 One Zone-IA, S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval, atau S3 Glacier Deep Archive sebelum melewati target penyimpan an minimum, Anda akan membayar biaya penghapusan awal untuk objek tersebut.
Manajemen penyimpanan	Anda membayar untuk fitur manajemen penyimpanan (Inventaris Amazon S3, analitik, dan penandaan objek) yang diaktifkan di bucket akun Anda.
Bandwidth	Anda membayar untuk semua bandwidth ke dan dari Amazon S3, kecuali untuk: • Data yang ditransfer dari internet • Data ditransfer ke instans Amazon Elastic Compute Cloud (Amazon EC2), jika instans sama dengan bucket S3 Wilayah AWS • Data ditransfer ke Amazon CloudFront (CloudFront) Anda juga membayar biaya untuk data apa pun yang ditransfer menggunakan Amazon S3 Transfer Acceleration.

Untuk informasi mendetail tentang biaya penggunaan Amazon S3 untuk penyimpanan, transfer data, dan layanan, lihat Harga Amazon S3 dan FAQ Amazon S3.

Untuk informasi tentang memahami kode dan singkatan yang digunakan dalam laporan penagihan dan penggunaan untuk Amazon S3, lihat. Memahami laporan AWS penagihan dan penggunaan Anda untuk Amazon S3

Info selengkapnya

- AWS laporan penggunaan untuk Amazon S3
- Menggunakan tag alokasi biaya bucket S3
- AWS Billing dan Manajemen Biaya
- Harga Amazon S3

AWS laporan penggunaan untuk Amazon S3

Saat mengunduh laporan penggunaan, Anda dapat memilih untuk menggabungkan data penggunaan berdasarkan jam, hari, atau bulan. Laporan penggunaan Amazon S3 mencantumkan operasi berdasarkan jenis penggunaan dan. Wilayah AWS Untuk laporan yang lebih mendetail tentang penggunaan penyimpanan Amazon S3 Anda, unduh laporan penggunaan AWS yang dibuat secara dinamis. Anda dapat memilih jenis penggunaan, operasi, dan periode waktu yang akan disertakan. Anda juga dapat memilih cara pengumpulan data. Untuk informasi selengkapnya tentang laporan penggunaan, lihat Laporan AWS Penggunaan di Panduan Pengguna Ekspor AWS Data.

Laporan penggunaan Amazon S3 mencakup informasi berikut:

- Layanan–Amazon S3
- Operasi–Operasi yang dilakukan pada bucket atau objek Anda. Untuk penjelasan terperinci tentang operasi Amazon S3, lihat Melacak Operasi dalam Laporan Penggunaan Anda.
- UsageType – Salah satu nilai berikut:
  - Kode yang mengidentifikasi jenis penyimpanan
  - Kode yang mengidentifikasi jenis permintaan
  - Kode yang mengidentifikasi jenis pengambilan
  - Kode yang mengidentifikasi jenis transfer data
  - Kode yang mengidentifikasi penghapusan awal dari penyimpanan S3 Intelligent-Tiering, S3 Standard-IA, S3 One Zone-Infrequent Access (S3 One Zone-IA), S3 Glacier Flexible Retrieval, atau penyimpanan S3 Glacier Deep Archive
  - `StorageObjectCount`–Jumlah objek yang disimpan dalam bucket tertentu

Untuk penjelasan terperinci tentang jenis penggunaan Amazon S3, lihat Memahami laporan AWS penagihan dan penggunaan Anda untuk Amazon S3.

- Sumber Daya–Nama bucket yang terkait dengan penggunaan yang terdaftar.
- StartTime— Waktu mulai hari penggunaan berlaku, di Coordinated Universal Time (UTC).
- EndTime— Waktu akhir hari penggunaan berlaku, di Coordinated Universal Time (UTC).
- UsageValue— Salah satu nilai volume berikut. Unit pengukuran yang umum untuk data adalah gigabyte (GB). Namun, tergantung pada layanan dan laporannya, terabyte (TB) mungkin juga akan digunakan.
  - Jumlah permintaan selama periode waktu yang ditentukan
  - Jumlah data yang ditransfer
  - Jumlah data yang disimpan dalam jam tertentu
  - Jumlah data yang terkait dengan restorasi dari penyimpanan S3 Standard-IA, S3 One Zone-IA, S3 Glacier Flexible Retrieval, atau S3 Glacier Deep Archive

> **Tip**
>
> Untuk informasi terperinci tentang setiap permintaan yang diterima Amazon S3 untuk objek Anda, aktifkan logging akses server untuk bucket Anda. Untuk informasi selengkapnya, lihat Pencatatan permintaan dengan pencatatan akses server.

Anda dapat mengunduh laporan penggunaan dalam bentuk file XML atau file nilai yang dipisahkan oleh koma (CSV). Berikut ini adalah contoh laporan penggunaan CSV yang dibuka di aplikasi spreadsheet.

Service	Operation	UsageType	Resource	StartTime	EndTime	UsageValue
AmazonS3	HeadBucket	USW2-C3DataTransfer-Out-Bytes	admin-created3	6/1/2017 0:00	7/1/2017 0:00	15309
AmazonS3	PutObject	USW2-C3DataTransfer-In-Bytes	admin-created3	6/1/2017 0:00	7/1/2017 0:00	19062
AmazonS3	HeadBucket	USW2-Requests-Tier2	admin-created3	6/1/2017 0:00	7/1/2017 0:00	68
AmazonS3	PutObjectForRepl	USW1-Requests-SIA-Tier1	ca-example-bucket	6/1/2017 0:00	7/1/2017 0:00	178294
AmazonS3	PutObjectForRepl	USW1-USW2-AWS-In-Bytes	ca-example-bucket	6/1/2017 0:00	7/1/2017 0:00	387929083
AmazonS3	GetObjectForRepl	USW2-Requests-NoCharge	admin-created3	6/1/2017 0:00	7/1/2017 0:00	108
AmazonS3	GetObjectForRepl	USW2-USW1-AWS-Out-Bytes	my-test-bucket-bash	6/1/2017 0:00	7/1/2017 0:00	387910021

Untuk informasi selengkapnya, lihat Memahami laporan AWS penagihan dan penggunaan Anda untuk Amazon S3.

Mengunduh Laporan AWS Penggunaan

Anda dapat mengunduh laporan penggunaan sebagai file XML/CSV.

Untuk mengunduh laporan penggunaan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di bilah judul, pilih nama pengguna atau ID akun Anda, lalu pilih Billing and Cost Management.
3. Di panel navigasi, pilih Laporan biaya dan penggunaan.
4. Di bawah Laporan AWS Penggunaan, pilih Buat Laporan Penggunaan.
5. Pada halaman Laporan penggunaan Unduh, pilih pengaturan berikut:
   - Layanan - Pilih Layanan Penyimpanan Sederhana Amazon.
   - Jenis Penggunaan–Untuk penjelasan detail tentang jenis penggunaan Amazon S3, lihat Memahami laporan AWS penagihan dan penggunaan Anda untuk Amazon S3.
   - Operasi–Untuk penjelasan detail tentang operasi Amazon S3, lihat Melacak Operasi dalam Laporan Penggunaan Anda.
   - Periode Waktu–Periode waktu yang ingin dicakup dalam laporan.
   - Granularitas Laporan–Apakah Anda ingin agar laporan menyertakan subtotal berdasarkan jam, hari, atau bulan.
6. Pilih Unduh, pilih format unduhan (Laporan XHTML atau Laporan CSV), lalu ikuti petunjuk untuk membuka atau menyimpan laporan.

Info selengkapnya

- Memahami laporan AWS penagihan dan penggunaan Anda untuk Amazon S3
- AWS Billing laporan untuk Amazon S3

Memahami laporan AWS penagihan dan penggunaan Anda untuk Amazon S3

> **Important**
>
> Pada 13 Mei 2024, kami mulai menerapkan perubahan untuk menghilangkan biaya atas permintaan tidak sah yang tidak diprakarsai oleh pemilik bucket. Setelah penerapan perubahan ini selesai, pemilik bucket tidak akan pernah dikenakan biaya permintaan atau bandwidth untuk permintaan yang mengembalikan kesalahan `AccessDenied` (HTTP`403 Forbidden`) saat permintaan ini dimulai dari luar AWS akun atau organisasi masing-masing. AWS Untuk informasi selengkapnya tentang daftar lengkap HTTP `3XX` dan kode `4XX` status yang tidak akan ditagih, lihatPenagihan untuk respons kesalahan Amazon S3. Perubahan penagihan ini tidak memerlukan pembaruan untuk aplikasi Anda dan berlaku untuk semua bucket S3. Ketika penerapan perubahan ini selesai di semua Wilayah AWS, kami akan memperbarui dokumentasi kami.

Laporan penagihan dan penggunaan Amazon S3 menggunakan kode dan singkatan. Untuk jenis penggunaan dalam tabel berikut, ganti, *`regionregion1`*, dan *`region2`* dengan singkatan dari daftar ini:

- APE1: Asia Pasifik (Hong Kong)
- APN1: Asia Pasifik (Tokyo)
- APN2: Asia Pasifik (Seoul)
- APN3: Asia Pasifik (Osaka)
- APS1: Asia Pasifik (Singapura)
- APS2: Asia Pasifik (Sydney)
- APS3: Asia Pasifik (Mumbai)
- APS4: Asia Pasifik (Jakarta)
- APS5: Asia Pasifik (Hyderabad)
- APS6: Asia Pasifik (Melbourne)
- CAN1: Kanada (Pusat)
- CAN2: Kanada Barat (Calgary)
- CNN1: Tiongkok (Beijing)

- CNW1: Tiongkok (Ningxia)
- AFS1: Afrika (Cape Town)
- EUC2: Eropa (Zürich)
- EUN1: Eropa (Stockholm)
- EUS2: Eropa (Spanyol)
- EUC1: Eropa (Frankfurt)
- EU: Eropa (Irlandia)
- EUS1: Eropa (Milan)
- EUW2: Eropa (London)
- EUW3: Eropa (Paris)
- ILC1: Israel (Tel Aviv)
- MEC1: Timur Tengah (UEA)
- MES1: Timur Tengah (Bahrain)
- SAE1: Amerika Selatan (Sao Paulo)
- UGW1: AWS GovCloud (AS-Barat)
- UGE1: AWS GovCloud (AS-Timur)
- USE1 (atau tanpa prefiks): AS Timur (Virginia Utara)
- USE2: AS Timur (Ohio)
- USW1: AS Barat (California Utara)
- USW2: AS Barat (Oregon)

Untuk tipe penggunaan S3 Multi-Region Access Points dalam tabel berikut, ganti *`regiongroup1`* dan *`regiongroup2`* dengan singkatan dari daftar ini:

- AP: Asia Pasifik
- AU: Australia
- EU: Eropa
- IN: India
- NA: Amerika Utara
- SA: Amerika Selatan

Kelompok wilayah adalah pengelompokan geografis dari beberapa. Wilayah AWS Untuk informasi selengkapnya, lihat Wilayah dan Zona Ketersediaan. Untuk informasi tentang harga menurut Wilayah AWS, lihat Harga Amazon S3.

Kolom pertama dalam tabel berikut mencantumkan jenis penggunaan yang muncul dalam laporan penagihan dan penggunaan Anda. Unit pengukuran yang umum untuk data adalah gigabyte (GB). Namun, tergantung pada layanan dan laporannya, terabyte (TB) mungkin juga akan digunakan.

Jenis Penggunaan

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region1-*region2*-AWS-In-ABytes	GB	Per Jam	Jumlah data yang dipercepat yang ditransfer ke *region1* dari *region2*
region1-*region2*-AWS-In-ABytes-T1	GB	Per Jam	Jumlah data akselerasi T1 yang ditransfer ke *region1* dari*region2*, di mana T1 mengacu pada CloudFront permintaan ke titik kehadiran (POPs) di Amerika Serikat, Eropa, dan Jepang
region1-*region2*-AWS-In-ABytes-T2	GB	Per Jam	Jumlah data akselerasi T2 yang ditransfer ke *region1* dari*region2*, di mana T2 mengacu pada CloudFront permintaan POPs di semua lokasi tepi lainnya AWS

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region1-*region2*-AWS-In-Bytes	GB	Per Jam	Jumlah data yang ditransfer *region1* dari *region2*
region1-*region2*-AWS-Out-ABytes	GB	Per Jam	Jumlah data yang dipercepat yang ditransfer dari *region1* ke *region2*
region1-*region2*-AWS-Out-ABytes-T1	GB	Per Jam	Jumlah data akselerasi T1 yang ditransfer dari *region1* ke*region2*, di mana T1 mengacu pada CloudFront permintaan ke POP di Amerika Serikat, Eropa, dan Jepang
region1-*region2*-AWS-Out-ABytes-T2	GB	Per Jam	Jumlah data akselerasi T2 yang ditransfer dari *region1* ke*region2*, di mana T2 mengacu pada CloudFront permintaan ke POP di semua lokasi tepi lainnya AWS
region1-*region2*-AWS-Out-Bytes	GB	Per Jam	Jumlah data yang ditransfer dari *region1* ke *region2*
region-BatchOperations-Jobs	Hitung	Per Jam	Jumlah pekerjaan Operasi Batch S3 yang dijalankan

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-BatchOperations-Objects	Hitung	Per Jam	Jumlah operasi objek yang dilakukan oleh Operasi Batch S3
region-Bulk-Retrieval-Bytes	GB	Per Jam	Jumlah data yang diambil dengan permintaan Massal S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive
region-BytesDeleted-GDA	GB	Bulanan	Jumlah data yang dihapus oleh `DeleteObject` operasi dari penyimpanan S3 Glacier Deep Archive
region-BytesDeleted-GIR	GB	Bulanan	Jumlah data yang dihapus oleh `DeleteObject` operasi dari penyimpanan S3 Glacier Instant Retrieval.
region-BytesDeleted-GLACIER	GB	Bulanan	Jumlah data yang dihapus oleh `DeleteObject` operasi dari penyimpanan S3 Glacier Flexible Retrieval
region-BytesDeleted-INT	GB	Bulanan	Jumlah data yang dihapus oleh `DeleteObject` operasi dari penyimpanan S3 Intelligent-Tiering

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-BytesDeleted-RRS	GB	Bulanan	Jumlah data yang dihapus oleh `DeleteObject` operasi dari penyimpanan Reduced Redundancy Storage (RRS)
region-BytesDeleted-SIA	GB	Bulanan	Jumlah data yang dihapus oleh `DeleteObject` operasi dari penyimpanan IA standar S3
region-BytesDeleted-STANDARD	GB	Bulanan	Jumlah data yang dihapus oleh `DeleteObject` operasi dari penyimpanan Standar S3
region-BytesDeleted-ZIA	GB	Bulanan	Jumlah data yang dihapus oleh `DeleteObject` operasi dari penyimpanan IA Zona Satu S3
region-C3DataTransfer-In-Bytes	GB	Per Jam	Jumlah data yang ditransfer ke Amazon S3 dari Amazon EC2 dalam waktu yang sama Wilayah AWS
region-C3DataTransfer-Out-Bytes	GB	Per Jam	Jumlah data yang ditransfer dari Amazon S3 ke Amazon EC2 di dalam Wilayah AWS yang sama

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-CloudFront-In-Bytes	GB	Per Jam	Jumlah data yang ditransfer Wilayah AWS ke suatu CloudFront distribusi
region-CloudFront-Out-Bytes	GB	Per Jam	Jumlah data yang ditransfer dari CloudFront distribusi Wilayah AWS ke suatu
region-DataTransfer-In-Bytes	GB	Per Jam	Jumlah data yang ditransfer ke Amazon S3 dari internet
region-DataTransfer-Out-Bytes	GB	Per Jam	Jumlah data yang ditransfer dari Amazon S3 ke internet[1]
region-DataTransfer-Regional-Bytes	GB	Per Jam	Jumlah data yang ditransfer dari Amazon S3 ke AWS sumber daya yang sama Wilayah AWS
region-EarlyDelete-ByteHrs	GB-Jam	Per Jam	Penggunaan penyimpanan prorata untuk objek yang dihapus dari penyimpanan S3 Glacier Flexible Retrieval sebelum komitmen minimum 90 hari berakhir[2]

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-EarlyDelete-GDA	GB-Jam	Per Jam	Penggunaan penyimpanan prorata untuk objek yang dihapus dari penyimpanan S3 Glacier Deep Archive sebelum komitmen minimum 180 hari berakhir [2]
region-EarlyDelete-GIR	GB-Jam	Per Jam	Penggunaan penyimpanan prorata untuk objek yang dihapus dari S3 Glacier Instant Retrieval sebelum komitmen minimum 90 hari berakhir.
region-EarlyDelete-GIR-SmObjects	GB-Jam	Per Jam	Penggunaan penyimpanan prorata untuk objek kecil (lebih kecil dari 128 KB) yang dihapus dari S3 Glacier Instant Retrieval sebelum komitmen minimum 90 hari berakhir.
region-EarlyDelete-SIA	GB-Jam	Per Jam	Penggunaan penyimpanan prorata untuk objek yang dihapus dari S3 Standard-IA sebelum komitmen minimum 30 hari berakhir[3]

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-EarlyDelete-SIA-SmObjects	GB-Jam	Per Jam	Penggunaan penyimpanan prorata untuk objek kecil (lebih kecil dari 128 KB) yang dihapus dari S3 Standard-IA sebelum komitmen minimum 30 hari berakhir[3]
region-EarlyDelete-ZIA	GB-Jam	Per Jam	Penggunaan penyimpanan prorata untuk objek yang dihapus dari S3 One Zone-IA sebelum komitmen minimum 30 hari berakhir[3]
region-EarlyDelete-ZIA-SmObjects	GB-Jam	Per Jam	Penggunaan penyimpanan prorata untuk objek kecil (lebih kecil dari 128 KB) yang dihapus dari S3 One Zone-IA sebelum komitmen minimum 30 hari berakhir[3]
region-Expedited-Retrieval-Bytes	GB	Per Jam	Jumlah data yang diambil dengan permintaan S3 Glacier Flexible Retrieval yang Dipercepat
region-Inventory-Objects Listed	Objek	Per Jam	Jumlah objek yang tercantum untuk grup objek (objek dikelompokkan berdasarkan bucket atau prefiks) dengan daftar inventaris

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-Monitoring-Automation-INT	Objek	Per Jam	Jumlah objek unik yang dipantau dan disusun secara otomatis dalam kelas penyimpanan S3 Intelligent-Tiering
region-MRAP-Out-Bytes	GB	Per Jam	Jumlah data yang ditransfer melalui titik akhir Titik Akses Multi-Wilayah S3 dari bucket di Wilayah (harga perutean data MRAP).
region-MRAP-In-Bytes	GB	Per Jam	Jumlah data yang ditransfer melalui titik akhir Titik Akses Multi-Wilayah S3 dari bucket di Wilayah (harga perutean data MRAP).
regiongroup1-regiongroup2- -MRAP-Out-Bytes	GB	Per Jam	Jumlah data yang ditransfer melalui titik akhir S3 Multi-Region Access Points dari bucket in *regiongroup1* ke klien yang *regiongroup2* berada di luar jaringan. AWS

Jenis Penggunaan	Unit	Granularitas	Deskripsi
regiongroup1-regiongroup2- - MRAP-In-Bytes	GB	Per Jam	Jumlah data yang ditransfer melalui titik akhir S3 Multi-Region Access Points ke bucket in *regiongroup1* dari klien yang *regiongroup2* berada di luar jaringan. AWS
region-OverwriteBytes-Copy-GDA	GB	Bulanan	Jumlah data yang ditimpa oleh CopyObject operasi dari penyimpanan S3 Glacier Deep Archive
region-OverwriteBytes-Copy-GIR	GB	Bulanan	Jumlah data yang ditimpa oleh CopyObject operasi dari penyimpanan S3 Glacier Instant Retrieval.
region-OverwriteBytes-Copy-GLACIER	GB	Bulanan	Jumlah data yang ditimpa oleh CopyObject operasi dari penyimpanan S3 Glacier Flexible Retrieval
region-OverwriteBytes-Copy-INT	GB	Bulanan	Jumlah data yang ditimpa oleh CopyObject operasi dari penyimpanan S3 Intelligent-Tiering

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-OverwriteBytes-Copy-RRS	GB	Bulanan	Jumlah data yang ditimpa oleh CopyObject operasi dari penyimpanan Reduced Redundancy Storage (RRS)
region-OverwriteBytes-Copy-SIA	GB	Bulanan	Jumlah data yang ditimpa oleh CopyObject operasi dari penyimpanan IA standar S3
region-OverwriteBytes-Copy-STANDARD	GB	Bulanan	Jumlah data yang ditimpa oleh CopyObject operasi dari penyimpanan Standar S3
region-OverwriteBytes-Copy-ZIA	GB	Bulanan	Jumlah data yang ditimpa oleh CopyObject operasi dari penyimpanan IA Zona Satu S3
region-OverwriteBytes-Put-GDA	GB	Bulanan	Jumlah data yang ditimpa oleh PutObject operasi dari penyimpanan S3 Glacier Deep Archive
region-OverwriteBytes-Put-GIR	GB	Bulanan	Jumlah data yang ditimpa oleh PutObject operasi dari penyimpanan S3 Glacier Instant Retrieval.

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-OverwriteBytes-Put-GLACIER	GB	Bulanan	Jumlah data yang ditimpa oleh PutObject operasi dari penyimpanan S3 Glacier Flexible Retrieval
region-OverwriteBytes-Put-INT	GB	Bulanan	Jumlah data yang ditimpa oleh PutObject operasi dari penyimpanan S3 Intelligent-Tiering
region-OverwriteBytes-Put-RRS	GB	Bulanan	Jumlah data yang ditimpa oleh PutObject operasi dari penyimpanan Reduced Redundancy Storage (RRS)
region-OverwriteBytes-Put-SIA	GB	Bulanan	Jumlah data yang ditimpa oleh PutObject operasi dari penyimpanan IA standar S3
region-OverwriteBytes-Put-STANDARD	GB	Bulanan	Jumlah data yang ditimpa oleh PutObject operasi dari penyimpanan Standar S3
region-OverwriteBytes-Put-ZIA	GB	Bulanan	Jumlah data yang ditimpa oleh PutObject operasi dari penyimpanan IA Zona Satu S3

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region1-*region2*-S3RTC-In-Bytes	GB	Bulanan	Jumlah data yang ditransfer untuk S3 Replication Time Control (S3 RTC) dari *region2* ke *region1* olehPutObject ReplTime ,,,, GetObject ReplTime InitiateMultipartUploadReplTime , UploadPartReplTime dan operasi CompleteMultipartUploadReplTime WriteACLReplTime
region1-*region2*-S3RTC-Out-Bytes	GB	Bulanan	Jumlah data yang ditransfer untuk S3 Replication Time Control (S3 RTC) dari *region1* ke *region2* olehPutObject ReplTime ,,,, GetObject ReplTime InitiateMultipartUploadReplTime , UploadPartReplTime dan operasi CompleteMultipartUploadReplTime WriteACLReplTime

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-Requests-GDA-Tier1	Hitung	Per Jam	JumlahPUT,,,, COPY POST CreateMultipartUpload UploadPart , atau CompleteMultipartUpload permintaan pada objek S3 Glacier Deep Archive 6
region-Requests-GDA-Tier2	Hitung	Per Jam	Jumlah GET dan HEAD permintaan pada objek S3 Glacier Deep Archive
region-Requests-GDA-Tier3	Hitung	Per Jam	Jumlah permintaan pemulihan standar S3 Glacier Deep Archive
region-Requests-GDA-Tier5	Hitung	Per Jam	Jumlah permintaan pemulihan Massal S3 Glacier Deep Archive
region-Requests-GIR-Tier1	Hitung	Per Jam	JumlahPUT,COPY, atau POST permintaan pada objek Pengambilan Instan S3 Glacier.
region-Requests-GIR-Tier2	Hitung	Per Jam	Jumlah GET dan semua permintaan Non-S3 Glacier Instant Retrieval -Tier1 lainnya pada objek S3 Glacier Instant Retrieval.

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-Requests-GLACIER-Tier1	Hitung	Per Jam	JumlahPUT,,,, COPY POST CreateMultipartUpload UploadPart , atau CompleteMultipartUpload permintaan pada objek Pengambilan Fleksibel S3 Glacier 6
region-Requests-GLACIER-Tier2	Hitung	Per Jam	Jumlah GET dan semua permintaan lainnya yang tidak tercantum pada objek Pengambilan Fleksibel S3 Glacier
region-Requests-INT-Tier1	Hitung	Per Jam	JumlahPUT,COPY, atau POST permintaan pada objek S3 Intelligent-Tiering
region-Requests-INT-Tier2	Hitung	Per Jam	Jumlah GET dan semua permintaan non-Tier1 lainnya untuk objek S3 Intelligent-Tiering
region-Requests-SIA-Tier1	Hitung	Per Jam	JumlahPUT,COPY, atau POST permintaan pada objek IA standar S3
region-Requests-SIA-Tier2	Hitung	Per Jam	Jumlah GET dan semua permintaan Non-S3 Glacier Instant Retrieval-Tier1 lainnya pada objek IA Standar S3

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-Requests-Tier1	Hitung	Per Jam	JumlahPUT,COPY, atau POST permintaan untuk Standar S3, RRS, dan tag, ditambah LIST permintaan untuk semua ember dan objek
region-Requests-Tier2	Hitung	Per Jam	Jumlah GET dan semua permintaan non-Tier1 lainnya
region-Requests-Tier3	Hitung	Per Jam	Jumlah permintaan siklus hidup ke S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive dan permintaan standar S3 Glacier Flexible Retrieval
region-Requests-Tier4	Hitung	Per Jam	Jumlah transisi siklus hidup ke penyimpanan S3 Glacier Instant, S3 Intelligent-Tiering, S3 Standard-IA, atau S3 One Zone-IA
region-Requests-Tier5	Hitung	Per Jam	Jumlah permintaan pemulihan Massal S3 Glacier Flexible Retrieval
region-Requests-Tier6	Hitung	Per Jam	Jumlah permintaan pemulihan S3 Glacier Fleksibel Dipercepat
region-Requests-Tier8	Hitung	Per Jam	Jumlah permintaan Hibah Akses S3

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-Requests-XZ-Tier1	Hitung	Per Jam	Jumlah PUT atau COPY permintaan pada objek S3 Express One Zone
region-Requests-XZ-Tier2	Hitung	Per Jam	Jumlah GET dan semua permintaan non-S3 Express One Zone-Tier 1 lainnya pada objek S3 Express One Zone
region-Requests-ZIA-Tier1	Hitung	Per Jam	JumlahPUT,COPY, atau POST permintaan pada objek IA Zona Satu S3
region-Requests-ZIA-Tier2	Hitung	Per Jam	Jumlah GET dan semua permintaan non-S3 One Zone-IA-Tier1 lainnya pada objek IA Zona Satu S3
region-Retrieval-GIR	GB	Per Jam	Jumlah data yang diambil dari penyimpanan S3 Glacier Instant Retrieval.
region-Retrieval-SIA	GB	Per Jam	Jumlah data yang diambil dari penyimpanan S3 Standard-IA
region-Retrieval-XZ	GB	Per Jam	Porsi data yang melebihi 512 KB dalam permintaan pengambilan tertentu (PUTatauCOPY) dengan penyimpanan S3 Express One Zone

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-Retrieval-ZIA	GB	Per Jam	Jumlah data yang diambil dari penyimpanan S3 One Zone-IA
region-S3DSSE-In-Bytes	GB	Bulanan	Jumlah data yang dienkripsi ganda oleh Amazon S3
region-S3DSSE-Out-Bytes	GB	Bulanan	Jumlah data terenkripsi ganda yang didekripsi oleh Amazon S3
region-S3G-DataTransfer-In-Bytes	GB	Per Jam	Jumlah data yang ditransfer ke Amazon S3 untuk memulihkan objek dari penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive
region-S3G-DataTransfer-Out-Bytes	GB	Per Jam	Jumlah data yang ditransfer dari Amazon S3 ke objek transisi ke penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive
region-Select-Returned-Bytes	GB	Per Jam	Jumlah data yang dikembalikan dengan permintaan Pilihan dari penyimpanan S3 Standard

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-Select-Returned-GIR-Bytes	GB	Per Jam	Jumlah data yang dikembalikan dengan permintaan Pilihan dari penyimpanan S3 Glacier Instant Retrieval.
region-Select-Returned-INT-Bytes	GB	Per Jam	Jumlah data yang dikembalikan dengan permintaan Pilihan dari penyimpanan S3 Intelligent-Tiering
region-Select-Returned-SIA-Bytes	GB	Per Jam	Jumlah data yang dikembalikan dengan permintaan Pilihan dari penyimpanan S3 Standard-IA
region-Select-Returned-ZIA-Bytes	GB	Per Jam	Jumlah data yang dikembalikan dengan permintaan Pilihan dari penyimpanan S3 One Zone-IA
region-Select-Scanned-Bytes	GB	Per Jam	Jumlah data yang dipindai dengan permintaan Pilihan dari penyimpanan S3 Standard
region-Select-Scanned-GIR-Bytes	GB	Per Jam	Jumlah data yang dipindai dengan permintaan Pilihan dari penyimpanan S3 Glacier Instant Retrieval.

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-Select-Scanned-INT-Bytes	GB	Per Jam	Jumlah data yang dipindai dengan permintaan Pilihan dari penyimpanan S3 Intelligent-Tiering
region-Select-Scanned-SIA-Bytes	GB	Per Jam	Jumlah data yang dipindai dengan permintaan Pilihan dari penyimpanan S3 Standard-IA
region-Select-Scanned-ZIA-Bytes	GB	Per Jam	Jumlah data yang dipindai dengan permintaan Pilihan dari penyimpanan S3 One Zone-IA
region-Standard-Retrieval-Bytes	GB	Per Jam	Jumlah data yang diambil dengan permintaan standar S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive
region-StorageAnalytics-ObjCount	Objek	Per Jam	Jumlah objek unik yang dipantau di setiap konfigurasi Analisis Kelas Penyimpanan.
region-StorageLens-ObjCount	Objek	Harian	Jumlah objek unik di setiap dasbor Lensa Penyimpanan S3 yang dilacak oleh metrik dan rekomendasi lanjutan Lensa Penyimpanan S3.

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-StorageLensFreeTier-ObjCount	Objek	Harian	Jumlah objek unik di setiap dasbor Lensa Penyimpanan S3 yang dilacak oleh metrik penggunaan Lensa Penyimpanan S3.
StorageObjectCount	Hitung	Harian	Jumlah objek yang disimpan dalam bucket tertentu
region-TagStorage-TagHrs	Tanda-Jam	Harian	Total tanda pada semua objek dalam bucket yang dilaporkan per jam
region-TimedStorage-ByteHrs	GB-Bulan	Harian	Jumlah GB-bulan bahwa data disimpan dalam penyimpanan Standar S3
region-TimedStorage-GDA-ByteHrs	GB-Bulan	Harian	Jumlah GB-bulan data disimpan di penyimpanan S3 Glacier Deep Archive
region-TimedStorage-GDA-Staging	GB-Bulan	Harian	Jumlah GB-bulan yang data disimpan di penyimpanan pementasan S3 Glacier Deep Archive
region-TimedStorage-GIR-ByteHrs	GB-Bulan	Harian	Jumlah GB-bulan data yang disimpan dalam penyimpanan S3 Glacier Instant Retrieval.

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-TimedStorage-GIR-SmObjects	GB-Bulan	Harian	Jumlah GB-bulan yang benda-benda kecil (lebih kecil dari 128 KB) disimpan dalam penyimpanan S3 Glacier Instant Retrieval.
region-TimedStorage-GlacierByteHrs	GB-Bulan	Harian	Jumlah GB-bulan yang data disimpan dalam penyimpanan S3 Glacier Flexible Retrieval
region-TimedStorage-GlacierStaging	GB-Bulan	Harian	Jumlah GB-bulan yang data disimpan dalam penyimpanan pementasan S3 Glacier Flexible Retrieval
region-TimedStorage-INT-FA-ByteHrs	GB-Bulan	Harian	Jumlah GB-bulan bahwa data disimpan di tingkat Akses Sering dari penyimpanan S3 Intelligent-Tiering 5
region-TimedStorage-INT-IA-ByteHrs	GB-Bulan	Harian	Jumlah GB-bulan bahwa data disimpan di tingkat Akses Jarang dari penyimpanan S3 Intelligent-Tiering
region-TimedStorage-INT-AA-ByteHrs	GB-Bulan	Harian	Jumlah GB-bulan bahwa data disimpan di tingkat Akses Arsip penyimpanan S3 Intelligent-Tiering

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-TimedStorage-INT-AIA-ByteHrs	GB-Bulan	Harian	Jumlah GB-bulan bahwa data disimpan di tingkat Akses Instan Arsip penyimpanan S3 Intelligent-Tiering
region-TimedStorage-INT-DAA-ByteHrs	GB-Bulan	Harian	Jumlah GB-bulan data disimpan di tingkat Deep Archive Access dari penyimpanan S3 Intelligent-Tiering
region-TimedStorage-RRS-ByteHrs	GB-Bulan	Harian	Jumlah GB-bulan yang data disimpan dalam penyimpanan Reduced Redundancy Storage (RRS)
region-TimedStorage-SIA-ByteHrs	GB-Bulan	Harian	Jumlah GB-bulan bahwa data disimpan dalam penyimpanan IA standar S3
region-TimedStorage-SIA-SmObjects	GB-Bulan	Harian	Jumlah GB-bulan benda kecil (lebih kecil dari 128 KB) disimpan dalam penyimpanan IA standar S3 [4]
region-TimedStorage-XZ-ByteHrs	GB-Bulan	Harian	Jumlah GB-bulan data yang disimpan dalam penyimpanan S3 Express One Zone

Jenis Penggunaan	Unit	Granularitas	Deskripsi
region-TimedStorage-ZIA-ByteHrs	GB-Bulan	Harian	Jumlah GB-bulan data disimpan di penyimpanan IA Zona Satu S3
region-TimedStorage-ZIA-SmObjects	GB-Bulan	Harian	Jumlah GB-bulan benda kecil (lebih kecil dari 128 KB) disimpan di penyimpanan IA Zona Satu S3
region-Upload-XZ	GB	Per Jam	Jumlah data yang melebihi 512 KB dalam permintaan unggahan tertentu (PUTatauCOPY) dengan S3 Express One Zone

Catatan

1. Jika Anda menghentikan transfer sebelum prosesnya selesai, jumlah data yang ditransfer dapat melebihi jumlah data yang diterima aplikasi. Perbedaan ini dapat terjadi karena permintaan penghentian transfer tidak dapat dieksekusi secara instan, dan sejumlah data mungkin dalam perjalanan, menunggu eksekusi permintaan penghentian. Data dalam perjalanan ini ditagih sebagai data yang ditransfer “keluar.”
2. Ketika objek yang diarsipkan ke S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval, atau kelas penyimpanan S3 Glacier Deep Archive dihapus, ditimpa, atau dialihkan ke kelas penyimpanan yang berbeda sebelum komitmen penyimpanan minimum telah berlalu, yaitu 90 hari untuk S3 Glacier Instant Retrieval dan S3 Glacier Flexible Retrieval, atau 180 hari untuk S3 S3 Glacier Deep Archive, ada biaya prorata per gigabyte untuk hari-hari yang tersisa.
3. Untuk objek yang berada di S3 Standard-IA atau S3 One Zone-IA storage, ketika mereka dihapus, ditimpa, atau dialihkan ke kelas penyimpanan yang berbeda sebelum 30 hari, ada biaya prorata per gigabyte untuk hari-hari yang tersisa.

4. Untuk benda-benda kecil (lebih kecil dari 128 KB) yang berada di S3 Standard-IA atau S3 One Zone-IA storage, ketika mereka dihapus, ditimpa, atau dialihkan ke kelas penyimpanan yang berbeda sebelum 30 hari, ada biaya prorata per gigabyte untuk hari-hari yang tersisa.
5. Tidak ada ukuran objek minimum yang dapat ditagih untuk objek di kelas penyimpanan S3 Intelligent-Tiering. Objek yang lebih kecil dari 128 KB tidak dipantau atau memenuhi syarat untuk peningkatan otomatis. Objek yang lebih kecil selalu disimpan di tingkat Akses Sering S3 Intelligent-Tiering.
6. Saat Anda memulai`CreateMultipartUpload`,`UploadPart`, atau `UploadPartCopy` permintaan ke kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, permintaan akan ditagih dengan tarif permintaan Standar S3 hingga Anda menyelesaikan unggahan multipart. Setelah unggahan selesai, `CompleteMultipartUpload` permintaan tunggal ditagih dengan `PUT` tarif untuk penyimpanan S3 Glacier tujuan. Suku cadang unggahan multibagian yang sedang berlangsung untuk kelas penyimpanan Pengambilan Fleksibel Gletser S3 `PUT` ke S3 Glacier ditagih sebagai S3 Glacier Flexible Retrieval Staging Staging Storage dengan kecepatan penyimpanan Standar S3 hingga pengunggahan selesai. Demikian pula, bagian unggahan multibagian yang sedang berlangsung untuk kelas penyimpanan S3 Glacier Deep Archive `PUT` ke S3 Glacier Deep Archive ditagih sebagai S3 Glacier Deep Archive Staging Storage dengan kecepatan penyimpanan Standar S3 hingga unggahan selesai.
7. S3 Express One Zone menerapkan biaya tetap per permintaan untuk ukuran permintaan hingga 512 KB. Biaya tambahan per GB diterapkan untuk `PUT` permintaan dan `GET` permintaan untuk porsi permintaan lebih besar dari 512 KB.
8. Untuk informasi tentang fitur yang didukung untuk kelas penyimpanan S3 Express One Zone, lihat Fitur Amazon S3 tidak didukung oleh S3 Express One Zone.
9. Jenis penggunaan dengan unit yang ditagih dalam GB dihitung dalam byte dalam laporan penggunaan.
10. GB-Bulan diperoleh dengan mengambil jumlah total GB-jam, menggabungkannya selama sebulan, dan kemudian membaginya dengan jumlah jam di bulan itu. Untuk mempelajari lebih lanjut lihat, Pertanyaan yang Sering Diajukan: Bagaimana saya akan ditagih dan ditagih untuk penggunaan Amazon S3 saya?

> **Note**
>
> Secara umum, pemilik bucket S3 ditagih untuk permintaan dengan respons HTTP `200 OK` yang berhasil dan respons kesalahan `4XX` klien HTTP. Pemilik bucket tidak ditagih untuk respons kesalahan `5XX` server HTTP, seperti `503 Slow Down` kesalahan HTTP. Untuk

informasi selengkapnya tentang kode kesalahan S3 di bawah HTTP `3XX` dan kode `4XX` status yang tidak ditagih, lihat. Penagihan untuk respons kesalahan Amazon S3 Untuk informasi selengkapnya tentang biaya penagihan jika bucket Anda dikonfigurasi sebagai bucket Requester Pays, lihat. Cara kerja Pembayaran Pemohon

Melacak Operasi dalam Laporan Penggunaan Anda

Operasi menggambarkan tindakan yang diambil pada AWS objek atau bucket Anda berdasarkan jenis penggunaan yang ditentukan. Operasi ditunjukkan dengan kode yang cukup jelas, seperti `PutObject` atau `ListBucket`. Untuk melihat tindakan pada bucket Anda yang menghasilkan jenis penggunaan tertentu, gunakan kode ini. Saat membuat laporan penggunaan, Anda dapat memilih untuk menyertakan Semua Operasi, atau operasi tertentu, misalnya, `GetObject`, untuk dilaporkan.

Info selengkapnya

- AWS laporan penggunaan untuk Amazon S3
- AWS Billing laporan untuk Amazon S3
- Harga Amazon S3
- Amazon S3 FAQ

Penagihan untuk respons kesalahan Amazon S3

> **Important**
>
> Pada 13 Mei 2024, kami mulai menerapkan perubahan untuk menghilangkan biaya atas permintaan tidak sah yang tidak diprakarsai oleh pemilik bucket. Setelah penerapan perubahan ini selesai, pemilik bucket tidak akan pernah dikenakan biaya permintaan atau bandwidth untuk permintaan yang mengembalikan kesalahan `AccessDenied` (HTTP`403 Forbidden`) saat permintaan ini dimulai dari luar AWS akun atau organisasi masing-masing. AWS Halaman saat ini menampilkan daftar lengkap HTTP `3XX` dan kode `4XX` status yang tidak akan ditagih. Perubahan penagihan ini tidak memerlukan pembaruan untuk aplikasi Anda dan berlaku untuk semua bucket S3. Ketika penerapan perubahan ini selesai di semua Wilayah AWS, kami akan memperbarui dokumentasi kami.

Secara umum, pemilik bucket S3 ditagih untuk permintaan dengan respons HTTP `200 OK` yang berhasil dan respons kesalahan `4XX` klien HTTP. Pemilik bucket tidak ditagih untuk respons kesalahan `5XX` server HTTP, seperti `503 Slow Down` kesalahan HTTP. Untuk informasi selengkapnya tentang biaya penagihan jika bucket Anda dikonfigurasi sebagai bucket Requester Pays, lihat. Cara kerja Pembayaran Pemohon

Tabel berikut mencantumkan kode kesalahan tertentu di bawah HTTP `3XX` dan kode `4XX` status yang tidak ditagih. Untuk bucket yang dikonfigurasi dengan hosting situs web, permintaan yang berlaku dan biaya lainnya akan tetap berlaku saat S3 mengembalikan dokumen kesalahan khusus atau untuk pengalihan khusus.

> **Note**
>
> Untuk `AccessDenied` (HTTP`403 Forbidden`), S3 tidak membebankan biaya kepada pemilik bucket saat permintaan dimulai di luar AWS akun individu pemilik bucket atau organisasi pemilik bucket. AWS

Kode status HTTP	Kode kesalahan	Deskripsi kode kesalahan	
301 Dipindahkan Secara Permanen	PermanentRedirect	Bucket yang Anda coba akses harus ditangani menggunakan titik akhir yang ditentukan. Kirim semua permintaan future ke endpoint ini.	
	PermanentRedirectControlError	Operasi API yang Anda coba akses harus ditangani menggunakan titik akhir yang ditentukan. Kirim semua permintaan future ke endpoint ini.	

Kode status HTTP	Kode kesalahan	Deskripsi kode kesalahan
307 Pengalihan Sementara	TemporaryRedirect	Anda sedang dialihkan ke bucket saat server Domain Name System (DNS) sedang diperbarui.
400 Permintaan Buruk	AuthorizationHeaderMalformed	Header otorisasi yang Anda berikan tidak valid.
	AuthorizationQueryParametersError	Parameter kueri otorisasi yang Anda berikan tidak valid.
	ExpiredToken	Token yang disediakan telah kedaluwarsa.
	IllegalLocationConstraintException	Anda mencoba mengakses bucket dari Wilayah yang berbeda dari tempat bucket berada. Untuk menghindari kesalahan ini, gunakan `--region` opsi. Misalnya: `aws s3 cp awsexample.txt s3:// example-s3-bucket/ --region ap-east-1` .

Kode status HTTP	Kode kesalahan	Deskripsi kode kesalahan	
	InvalidArgument	Kesalahan ini dapat terjadi karena alasan berikut: • Argumen yang ditentukan tidak valid. • Permintaan itu tidak memiliki header yang diperlukan. • Argumen yang ditentukan tidak lengkap atau dalam format yang salah. • Argumen yang ditentukan harus memiliki panjang lebih besar dari atau sama dengan 3.	
	InvalidDigest	Nilai Content-MD5 atau checksum yang Anda tentukan tidak valid.	
	InvalidEncryptionAlgorithmError	Permintaan enkripsi yang Anda tentukan tidak valid. Nilai yang valid adalah`AES256`.	

Kode status HTTP	Kode kesalahan	Deskripsi kode kesalahan	
	InvalidRequest	Kesalahan ini dapat terjadi karena alasan berikut: • Permintaan menggunakan versi tanda tangan yang salah. Gunakan `AWS4-HMAC-SHA256` (Versi Tanda Tangan 4). • Titik akses hanya dapat dibuat untuk bucket yang ada. • Titik akses tidak dalam keadaan di mana ia dapat dihapus. • Titik akses hanya dapat dicantumkan untuk bucket yang ada. • Token berikutnya tidak valid. • Setidaknya satu tindakan harus ditentukan dalam aturan siklus hidup. •	

Kode status HTTP	Kode kesalahan	Deskripsi kode kesalahan	
		Setidaknya satu aturan siklus hidup harus ditentukan. • Jumlah aturan siklus hidup tidak boleh melebihi batas yang diizinkan dari 1000 aturan. • Rentang untuk `MaxResults` parameter tidak valid. • Permintaan SOAP harus dibuat melalui koneksi HTTPS. • Amazon S3 Transfer Acceleration tidak didukung untuk bucket dengan nama yang tidak sesuai dengan DNS. • Amazon S3 Transfer Acceleration tidak didukung untuk bucket dengan periode (.) dalam nama mereka. • Endpoint Amazon S3 Transfer Acceleration	

Kode status HTTP	Kode kesalahan	Deskripsi kode kesalahan	
		hanya mendukung permintaan gaya virtual. • Amazon S3 Transfer Acceleration tidak dikonfigurasi pada bucket ini. • Amazon S3 Transfer Acceleration dinonaktifkan di bucket ini. • Amazon S3 Transfer Acceleration tidak didukung pada bucket ini. Untuk bantuan, hubungi AWS Support. • Amazon S3 Transfer Acceleration tidak dapat diaktifkan di bucket ini. Untuk bantuan, hubungi AWS Support. • Nilai yang bertentangan disediakan dalam header HTTP dan parameter kueri. • Nilai yang bertentangan disediakan di header	

Kode status HTTP	Kode kesalahan	Deskripsi kode kesalahan
		HTTP dan bidang formulir POST. • CopyObject permintaan dibuat pada objek yang berukuran lebih besar dari 5GB.
	InvalidSoapRequest	Badan permintaan SOAP tidak valid.
	InvalidStorageClass	Kelas penyimpanan yang Anda tentukan tidak valid.
	InvalidTag	Permintaan Anda berisi input tag yang tidak valid. Misalnya, permintaan Anda mungkin berisi kunci duplikat, kunci, atau nilai yang terlalu panjang, atau tag sistem.
	InvalidToken	Token yang disediakan cacat atau tidak valid.
	InvaliDuri	URI yang ditentukan tidak dapat diuraikan.
	KeyTooLongError	Kunci Anda terlalu panjang.

Kode status HTTP	Kode kesalahan	Deskripsi kode kesalahan	
	MalformeDACIError	ACL yang Anda berikan tidak terbentuk dengan baik atau tidak memvalidasi skema kami yang diterbitkan.	
	MalformedPostRequest	Isi permintaan POST Anda tidak terbentuk dengan baik multipart/form-data.	
	MalformedXML	XHTML yang Anda berikan tidak terbentuk dengan baik atau tidak memvalidasi skema kami yang dipublikasikan.	
	MaxPostPreDataLengthExceededError	Bidang permintaan POST Anda sebelum file upload terlalu besar.	
	MetadataTooLarge	Header metadata Anda melebihi ukuran metadata maksimum yang diizinkan.	
	MissingRequestBodyError	Anda mengirim dokumen XHTML kosong sebagai permintaan.	
	MissingSecurityHeader	Permintaan Anda tidak memiliki header yang diperlukan.	

Kode status HTTP	Kode kesalahan	Deskripsi kode kesalahan	
	NoLoggingStatusForKey	Tidak ada yang namanya subresource status logging untuk kunci.	
	RequestHeaderSectionTooLarge	Header permintaan dan parameter kueri yang digunakan untuk membuat permintaan melebihi ukuran maksimum yang diizinkan	
	UnexpectedContent	Permintaan ini berisi konten yang tidak didukung.	
	UserKeyMustBeSpecified	Permintaan bucket POST harus berisi nama bidang yang ditentukan. Jika ditentukan, periksa urutan bidang.	
	IncorrectEndpoint	Bucket yang ditentukan ada di Wilayah lain. Permintaan langsung ke titik akhir yang benar.	
403 Dilarang	RequestTimeTooSkewed	Perbedaan antara waktu permintaan dan waktu server terlalu besar.	

Kode status HTTP	Kode kesalahan	Deskripsi kode kesalahan	
	SignatureDoesNotMatch	Tanda tangan permintaan yang dihitung server tidak sesuai dengan tanda tangan yang Anda berikan. Periksa kunci akses AWS rahasia Anda dan metode penandatanganan. Untuk informasi selengkapnya, lihat Otentikasi REST dan Otentikasi SOAP.	
	NotSignedUp	Akun Anda tidak mendaftar untuk layanan Amazon S3. Anda harus mendaftar sebelum dapat menggunakan Amazon S3. Anda dapat mendaftar di URL berikut: https://aws.amazon.com/s3	
	InvalidSecurity	Kredensi keamanan yang diberikan tidak valid.	
	InvalidPayer	Semua akses ke objek ini telah dinonaktifkan. Untuk bantuan lebih lanjut, lihat Hubungi Kami.	
	InvalidAccessKeyId	ID kunci AWS akses yang Anda berikan tidak ada dalam catatan kami.	

Kode status HTTP	Kode kesalahan	Deskripsi kode kesalahan
	AccountProblem	Ada masalah dengan Anda Akun AWS yang mencegah operasi selesai dengan sukses. Untuk bantuan lebih lanjut, lihat Hubungi Kami.
	UnauthorizedAccessError	Hanya berlaku di Wilayah China. Dikembalikan saat permintaan dibuat ke bucket yang tidak memiliki lisensi ICP. Untuk informasi lebih lanjut, lihat ICP Recordal.
404 Tidak Ditemukan	NoSuchUpload	Unggahan multipart yang ditentukan tidak ada. ID unggahan mungkin tidak valid, atau unggahan multibagian mungkin telah dibatalkan atau diselesaikan.
	NoSuchWebsiteConfiguration	Bucket yang ditentukan tidak memiliki konfigurasi situs web.
405 Metode Tidak Diizinkan	MethodNotAllowed	Metode yang ditentukan tidak diperbolehkan terhadap sumber daya ini.

Kode status HTTP	Kode kesalahan	Deskripsi kode kesalahan	
409 Konflik	BucketAlreadyExists	Nama bucket yang diminta tidak tersedia. Namespace bucket dibagikan oleh semua pengguna sistem. Tentukan nama yang berbeda dan coba lagi.	
	InvalidBucketState	Permintaan tidak valid untuk status bucket saat ini.	
	OperationAborted	Operasi kondisional yang bertentangan saat ini sedang berlangsung terhadap sumber daya ini. Coba lagi.	
411 Panjang Diperlukan	MissingContentLength	Anda harus menyediakan header HTTP Content-Length.	
412 Prasyarat Gagal	RequestIsNotMultiPartContent	Permintaan bucket POST harus dari multipart/form-data tipe enclosure.	

Menyaring dan mengambil data menggunakan Amazon S3 Select

Dengan Amazon S3 Select, Anda dapat menggunakan pernyataan bahasa kueri terstruktur (SQL) untuk memfilter konten objek Amazon S3 dan mengambil hanya subset data yang Anda butuhkan. Dengan menggunakan Amazon S3 Select untuk memfilter data ini, Anda dapat mengurangi jumlah data yang ditransfer oleh Amazon S3, sehingga mengurangi biaya dan latensi untuk pengambilan data ini.

Amazon S3 Select hanya memungkinkan Anda untuk menanyakan satu objek pada satu waktu. Ia bekerja pada objek yang disimpan dalam CSV, JSON, atau Apache Parquet format. Ia juga bekerja dengan objek yang dikompresi dengan GZIP atau BZIP2 (hanya untuk objek CSV dan JSON), dan objek terenkripsi sisi server. Anda dapat menentukan format hasil sebagai CSV atau JSON, dan Anda dapat menentukan pembatasan untuk catatan di dalam hasilnya.

Anda dapat mengajukan pemindahan ekspresi SQL ke Amazon S3. Amazon S3 Select mendukung squbset SQL. Untuk informasi selengkapnya tentang elemen SQL yang didukung oleh Amazon S3 Select, lihat Referensi SQL untuk Amazon S3 Select.

Anda dapat melakukan kueri SQL dengan menggunakan konsol Amazon S3,AWS CLI(), AWS Command Line Interface operasi REST API, `SelectObjectContent` atau SDK. AWS

> **Note**
>
> Konsol Amazon S3 membatasi jumlah data yang ditampilkan hingga 40 MB. Untuk mengambil lebih banyak data, gunakan AWS CLI atau API.

Persyaratan dan batasan

Berikut adalah persyaratan untuk menggunakan Amazon S3 Select:

- Anda harus memiliki izin `s3:GetObject` untuk objek yang Anda minta.
- Jika objek yang Anda tanyakan dienkripsi dengan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C), Anda harus menggunakan `https`, dan Anda harus menyediakan kunci enkripsi saat pengajuan.

Batasan berikut berlaku saat menggunakan Amazon S3 Select:

- S3 Select hanya dapat menanyakan satu objek per permintaan.
- Panjang maksimum ekspresi SQL adalah 256 KB.
- Panjang maksimum catatan dalam input atau hasil adalah 1 MB.
- Amazon S3 Select hanya dapat mengeluarkan data yang disimpan dengan format output JSON.
- Anda tidak dapat menanyakan objek yang disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval, S3 Glacier Deep Archive, atau Reduced Redundancy Storage (RRS). Anda juga tidak dapat menanyakan objek yang disimpan di tingkat Akses Arsip Tingkat Cerdas S3 atau tingkat

Akses Arsip Dalam Tingkat Cerdas S3. Untuk informasi selengkapnya tentang kelas penyimpanan, lihat Menggunakan kelas penyimpanan Amazon S3.

Batasan tambahan berlaku saat menggunakan Amazon S3 Select dengan Parquet objek:

- Amazon S3 Select hanya mendukung kompresi berbentuk kolom yang menggunakan GZIP atau Snappy. Amazon S3 Select tidak mendukung kompresi seluruh objek untuk objek. Parquet
- Amazon S3 Select tidak mendukung output Parquet. Anda harus menentukan format output sebagai CSV atau JSON.
- Ukuran maksimum untuk grup baris yang tidak dikompres adalah 512 MB.
- Anda harus menggunakan jenis data yang ditentukan di dalam skema objek.
- Pemilihan pada bidang berulang hanya akan menampilkan nilai terakhir.

Membuat permintaan

Saat Anda membuat sebuah permintaan, Anda harus menyediakan perincian objek yang sedang diminta dengan menggunakan objek `InputSerialization`. Anda dapat menyediakan perincian tentang cara menampilkan hasil dengan menggunakan objek `OutputSerialization`. Anda juga dapat menyertakan ekspresi SQL yang digunakan oleh Amazon S3 untuk memfilter permintaan.

Untuk informasi lebih lanjut tentang pembuatan permintaan Amazon S3 Select, lihat SelectObjectContent di Referensi API Amazon Simple Storage Service. Anda juga dapat melihat salah satu contoh kode SDK di bagian berikut.

Permintaan yang menggunakan rentang pemindaian

Dengan Amazon S3 Select, Anda dapat memindai subset sebuah objek dengan menentukan rentang byte ke kueri. Kemampuan ini memungkinkan Anda melakukan pemindaian pada keseluruhan objek secara paralel dengan membagi pekerjaan menjadi beberapa permintaan Amazon S3 Select yang terpisah untuk serangkaian rentang pemindaian non-tumpang tindih.

Rentang pemindaian tidak perlu diselaraskan dengan batasan catatan. Permintaan rentang pemindaian Amazon S3 Select dijalankan di seluruh rentang byte yang Anda tentukan. Catatan yang dimulai dengan rentang pemindaian yang ditentukan tetapi melampaui rentang pemindaian tersebut akan diproses oleh kueri. Contoh berikut menampilkan objek Amazon S3 yang berisi serangkaian catatan dengan format CSV yang dibatasi baris:

```
A,B
C,D
D,E
E,F
G,H
I,J
```

Misalkan Anda menggunakan parameter `ScanRange` Amazon S3 Select dan Start pada (Byte) 1 dan End pada (Byte) 4. Jadi, rentang pemindaian akan diawali di `","` dan pemindaian dilakukan hingga akhir catatan yang dimulai di `C`. Permintaan rentang pemindaian Anda akan menampilkan hasil `C, D` karena hasil tersebut adalah akhir catatannya.

Amazon S3 Pilih dukungan permintaan rentang pemindaianParquet, CSV (tanpa pembatas yang dikutip), atau objek JSON (hanya dalam mode). `LINES` Objek CSV dan JSON harus dalam kondisi tidak dikompres. Untuk objek CSV dan JSON berbasis baris, ketika rentang pemindaian ditentukan sebagai bagian dari permintaan Amazon S3 Select, semua catatan yang dimulai di dalam rentang pemindaian akan diproses. Untuk objek Parquet, semua grup baris yang dimulai di dalam rentang pemindaian yang diminta akan diproses.

Permintaan rentang pemindaian Amazon S3 Select tersedia untuk digunakan dengan AWS CLI, Amazon S3 API, dan SDK. AWS Anda dapat menggunakan parameter `ScanRange` di dalam permintaan Amazon S3 Select untuk fitur ini. Untuk informasi lebih lanjut, lihat SelectObjectContent di dalam Referensi API Amazon Simple Storage Service.

Kesalahan

Amazon S3 Select akan menampilkan kode kesalahan dan pesan kesalahan terkait saat ditemukannya masalah ketika suatu kueri sedang dijalankan. Untuk daftar kode dan deskripsi kesalahan, lihat bagian Daftar Kode Kesalahan SELECT Object Content dari halaman Respons Kesalahan di dalam Referensi API Amazon Simple Storage Service.

Untuk informasi lebih lanjut tentang Amazon S3 Select, lihat topik-topik berikut.

Topik

- Contoh menggunakan Amazon S3 Select pada objek
- Referensi SQL untuk Amazon S3 Select

Contoh menggunakan Amazon S3 Select pada objek

Anda dapat menggunakan S3 Select untuk memilih konten dari satu objek dengan menggunakan konsol Amazon S3, REST API, dan AWS SDK.

Untuk informasi lebih lanjut tentang fungsi SQL yang didukung untuk S3 Select, lihat Fungsi SQL.

Menggunakan konsol S3

Untuk memilih konten dari objek di konsol Amazon S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Pilih bucket yang berisi objek yang ingin Anda pilih kontennya, lalu pilih nama objeknya.
4. Pilih Tindakan objek, dan pilih Kueri dengan S3 Select.
5. Konfigurasikan Pengaturan input, berdasarkan format data input Anda.
6. Konfigurasikan Pengaturan output, berdasarkan format output yang ingin Anda terima.
7. Untuk mengekstrak catatan dari objek yang dipilih, di bawah kueri SQL, masukkan perintah SELECT SQL. Untuk informasi lebih lanjut tentang cara menulis perintah SQL, lihat Referensi SQL untuk Amazon S3 Select.
8. Setelah memasukkan kueri SQL, pilih Jalankan kueri SQL. Kemudian, di bawah Hasil kueri, Anda akan melihat hasil kueri SQL Anda.

Menggunakan API REST

Anda dapat menggunakan AWS SDK untuk memilih konten dari objek. Namun, jika aplikasi Anda memerlukannya, Anda dapat mengirimkan permintaan REST secara langsung. Untuk informasi lebih lanjut tentang permintaan dan format respons, kunjungi SelectObjectContent.

Menggunakan AWS SDK

Anda dapat menggunakan Amazon S3 Select untuk memilih beberapa konten objek dengan menggunakan metode ini`selectObjectContent`. Jika metode ini berhasil, Amazon S3 Select akan menampilkan ekspresi SQL.

Java

Kode Java berikut ini menampilkan nilai kolom pertama untuk setiap catatan yang disimpan di dalam objek yang mengandung data yang disimpan dengan format CSV. Kode ini juga meminta agar pesan `Progress` dan `Stats` dapat ditampilkan. Anda harus menyediakan nama bucket yang valid dan objek yang mengandung data dengan format CSV.

Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
package com.amazonaws;

import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CSVInput;
import com.amazonaws.services.s3.model.CSVOutput;
import com.amazonaws.services.s3.model.CompressionType;
import com.amazonaws.services.s3.model.ExpressionType;
import com.amazonaws.services.s3.model.InputSerialization;
import com.amazonaws.services.s3.model.OutputSerialization;
import com.amazonaws.services.s3.model.SelectObjectContentEvent;
import com.amazonaws.services.s3.model.SelectObjectContentEventVisitor;
import com.amazonaws.services.s3.model.SelectObjectContentRequest;
import com.amazonaws.services.s3.model.SelectObjectContentResult;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.concurrent.atomic.AtomicBoolean;

import static com.amazonaws.util.IOUtils.copy;

/**
 * This example shows how to query data from S3Select and consume the response in
 the form of an
 * InputStream of records and write it to a file.
 */

public class RecordInputStreamExample {

    private static final String BUCKET_NAME = "${my-s3-bucket}";
    private static final String CSV_OBJECT_KEY = "${my-csv-object-key}";
```

```
    private static final String S3_SELECT_RESULTS_PATH = "${my-s3-select-results-
path}";
    private static final String QUERY = "select s._1 from S3Object s";

    public static void main(String[] args) throws Exception {
        final AmazonS3 s3Client = AmazonS3ClientBuilder.defaultClient();

        SelectObjectContentRequest request = generateBaseCSVRequest(BUCKET_NAME,
 CSV_OBJECT_KEY, QUERY);
        final AtomicBoolean isResultComplete = new AtomicBoolean(false);

        try (OutputStream fileOutputStream = new FileOutputStream(new File
 (S3_SELECT_RESULTS_PATH));
             SelectObjectContentResult result =
 s3Client.selectObjectContent(request)) {
            InputStream resultInputStream =
 result.getPayload().getRecordsInputStream(
                    new SelectObjectContentEventVisitor() {
                        @Override
                        public void visit(SelectObjectContentEvent.StatsEvent event)
                        {
                            System.out.println(
                                    "Received Stats, Bytes Scanned: " +
 event.getDetails().getBytesScanned()
                                            +  " Bytes Processed: " +
 event.getDetails().getBytesProcessed());
                        }

                        /*
                         * An End Event informs that the request has finished
 successfully.
                         */
                        @Override
                        public void visit(SelectObjectContentEvent.EndEvent event)
                        {
                            isResultComplete.set(true);
                            System.out.println("Received End Event. Result is
 complete.");
                        }
                    }
            );

            copy(resultInputStream, fileOutputStream);
        }
```

```
        /*
         * The End Event indicates all matching records have been transmitted.
         * If the End Event is not received, the results may be incomplete.
         */
        if (!isResultComplete.get()) {
            throw new Exception("S3 Select request was incomplete as End Event was
 not received.");
        }
    }

    private static SelectObjectContentRequest generateBaseCSVRequest(String bucket,
 String key, String query) {
        SelectObjectContentRequest request = new SelectObjectContentRequest();
        request.setBucketName(bucket);
        request.setKey(key);
        request.setExpression(query);
        request.setExpressionType(ExpressionType.SQL);

        InputSerialization inputSerialization = new InputSerialization();
        inputSerialization.setCsv(new CSVInput());
        inputSerialization.setCompressionType(CompressionType.NONE);
        request.setInputSerialization(inputSerialization);

        OutputSerialization outputSerialization = new OutputSerialization();
        outputSerialization.setCsv(new CSVOutput());
        request.setOutputSerialization(outputSerialization);

        return request;
    }
}
```

JavaScript

Untuk JavaScript contoh yang menggunakan operasi AWS SDK for JavaScript with S3 `SelectObjectContent` API untuk memilih catatan dari file JSON dan CSV yang disimpan di Amazon S3, lihat posting blog Memperkenalkan dukungan untuk Amazon S3 Select di file. AWS SDK for JavaScript

Python

Untuk contoh Python tentang penggunaan kueri SQL untuk melakukan pencarian lewat data yang dimuat ke Amazon S3 sebagai file nilai terpisah-koma (CSV) dengan menggunakan S3 Select,

lihat postingan blog Meminta data tanpa server atau basis data dengan menggunakan Amazon S3 Select.

Referensi SQL untuk Amazon S3 Select

Referensi ini berisi deskripsi elemen bahasa kueri terstruktur (SQL) yang didukung oleh Amazon S3 Select.

Topik

- Perintah SELECT
- Jenis Data
- Operator
- Kata kunci yang dicadangkan
- Fungsi SQL

Perintah SELECT

Amazon S3 Select hanya mendukung perintah `SELECT` SQL. Klausul standar ANSI berikut didukung untuk `SELECT`:

- Daftar `SELECT`
- Klausul `FROM`
- Klausul `WHERE`
- Klausul `LIMIT`

> **Note**
> Kueri Amazon S3 Select saat ini tidak mendukung subkueri atau Joins.

Daftar SELECT

Daftar `SELECT` mencantumkan kolom, fungsi, dan ekspresi yang ingin Anda tampilkan dari kueri. Daftar ini mewakili output kueri.

```
SELECT *
SELECT projection1 AS column_alias_1, projection2 AS column_alias_2
```

Bentuk pertama dari `SELECT` dengan `*` (tanda bintang) mengembalikan setiap baris yang melewati `WHERE` klausul, apa adanya. Bentuk kedua dari `SELECT` membuat baris dengan ekspresi skalar hasil yang ditentukan pengguna ***projection1***dan ***projection2***untuk setiap kolom.

Klausul FROM

Amazon S3 Select mendukung bentuk `FROM` klausul berikut:

```
FROM table_name
FROM table_name alias
FROM table_name AS alias
```

Dalam setiap bentuk `FROM` klausa, `table_name` adalah `S3Object` yang sedang ditanyakan. Pengguna yang berasal dari basis data relasional tradisional dapat menganggap ini sebagai skema basis data yang berisi beberapa tampilan di atas tabel.

Mengikuti SQL standar, klausul`FROM` membuat baris yang difilter dalam klausul `WHERE` dan diproyeksikan dalam daftar `SELECT`.

Untuk objek JSON yang disimpan di Amazon S3 Select, Anda juga dapat menggunakan bentuk klausul `FROM` berikut:

```
FROM S3Object[*].path
FROM S3Object[*].path alias
FROM S3Object[*].path AS alias
```

Menggunakan bentuk klausul `FROM`, Anda dapat memilih dari arrays atau objek di dalam objek JSON. Anda dapat menentukan `path` dengan menggunakan salah satu bentuk berikut:

- Berdasarkan nama (dalam objek): `.name` atau `['name']`
- Berdasarkan indeks (dalam array): `[index]`
- Berdasarkan karakter wildcard (dalam objek): `.*`
- Berdasarkan karakter wildcard (dalam array): `[*]`

> **Note**
>
> - Bentuk klausul `FROM` ini hanya berfungsi dengan objek JSON.
> - Karakter wildcard selalu mengeluarkan setidaknya satu record. Jika tidak ada catatan yang cocok, Amazon S3 Select mengeluarkan nilai `MISSING`. Selama serialisasi output (setelah kueri selesai berjalan), Amazon S3 Select menggantikan `MISSING` nilai dengan catatan kosong.
> - Fungsi agregat (`AVG`, `COUNT`, `MAX`, `MIN`, dan `SUM`) melompati nilai `MISSING`.
> - Jika Anda tidak menyediakan alias saat menggunakan karakter wildcard, Anda dapat merujuk pada baris dengan menggunakan elemen terakhir di jalur tersebut. Misalnya, Anda dapat memilih semua harga dari daftar buku dengan menggunakan kueri`SELECT price FROM S3Object[*].books[*].price`. Jika jalan berakhir dengan karakter wildcard alih-alih nama, maka Anda dapat menggunakan nilai tersebut `_1` untuk merujuk pada baris. Misalnya, alih-alih `SELECT price FROM S3Object[*].books[*].price`, Anda dapat menggunakan kueri `SELECT _1.price FROM S3Object[*].books[*]`.
> - Amazon S3 Select selalu memperlakukan dokumen JSON sebagai serangkaian nilai tingkat akar. Jadi, bahkan jika objek JSON yang Anda kueri hanya memiliki satu elemen akar, klausa `FROM` harus dimulai dengan `S3Object[*]`. Namun, karena alasan kompatibilitas, Amazon S3 Select memungkinkan Anda menghilangkan karakter wildcard jika Anda tidak menyertakan jalur. Dengan demikian, klausa lengkap `FROM S3Object` setara dengan `FROM S3Object[*] as S3Object`. Jika Anda menyertakan jalan, Anda juga harus menggunakan karakter wildcard. Jadi, `FROM S3Object` dan `FROM S3Object[*].`*`path`* keduanya klausa yang valid, tetapi `FROM S3Object.`*`path`* tidak.

Example

Contoh:

Contoh #1

Contoh ini menampilkan hasil saat menggunakan set data dan kueri berikut:

```
{ "Rules": [ {"id": "1"}, {"expr": "y > x"}, {"id": "2", "expr": "z = DEBUG"} ]}
{ "created": "June 27", "modified": "July 6" }
```

```
SELECT id FROM S3Object[*].Rules[*].id
```

```
{"id":"1"}
{}
{"id":"2"}
{}
```

Amazon S3 Select menghasilkan setiap hasil karena alasan berikut:

- `{"id":"id-1"}`- `S3Object[0].Rules[0].id` menghasilkan korek api.
- `{}`— `S3Object[0].Rules[1].id` tidak cocok dengan catatan, jadi Amazon S3 Select mengeluarkan`MISSING`, yang kemudian diubah menjadi rekaman kosong selama serialisasi keluaran dan dikembalikan.
- `{"id":"id-2"}`- `S3Object[0].Rules[2].id` menghasilkan korek api.
- `{}`— `S3Object[1]` tidak cocok`Rules`, jadi Amazon S3 Select mengeluarkan`MISSING`, yang kemudian diubah menjadi rekaman kosong selama serialisasi keluaran dan dikembalikan.

Jika Anda tidak ingin Amazon S3 Select mengembalikan catatan kosong saat tidak menemukan kecocokan, Anda dapat menguji nilai `MISSING`. Kueri berikut mengembalikan hasil yang sama seperti kueri sebelumnya, tetapi dengan nilai-nilai kosong dihilangkan:

```
SELECT id FROM S3Object[*].Rules[*].id WHERE id IS NOT MISSING
```

```
{"id":"1"}
{"id":"2"}
```

Contoh #2

Contoh ini menampilkan hasil saat menggunakan set data dan kueri berikut:

```
{ "created": "936864000", "dir_name": "important_docs", "files": [ { "name": "." },
 { "name": ".." }, { "name": ".aws" }, { "name": "downloads" } ], "owner": "Amazon
 S3" }
{ "created": "936864000", "dir_name": "other_docs", "files": [ { "name": "." },
 { "name": ".." }, { "name": "my stuff" }, { "name": "backup" } ], "owner": "User" }
```

```
SELECT d.dir_name, d.files FROM S3Object[*] d
```

```
{"dir_name":"important_docs","files":[{"name":"."},{"name":".."},{"name":".aws"},
{"name":"downloads"}]}
{"dir_name":"other_docs","files":[{"name":"."},{"name":".."},{"name":"my stuff"},
{"name":"backup"}]}
```

```
SELECT _1.dir_name, _1.owner FROM S3Object[*]
```

```
{"dir_name":"important_docs","owner":"Amazon S3"}
{"dir_name":"other_docs","owner":"User"}
```

Klausul WHERE

Klausul `WHERE` mengikuti sintaks ini:

```
WHERE condition
```

`WHERE`Klausul memfilter baris berdasarkan. *`condition`* Kondisi adalah ekspresi yang memiliki hasil Boolean. Hanya baris yang kondisinya dievaluasi sebagai `TRUE` yang dikembalikan di hasilnya.

Klausul LIMIT

Klausul `LIMIT` mengikuti sintaks ini:

```
LIMIT number
```

`LIMIT`Klausul membatasi jumlah record yang Anda ingin kueri dikembalikan berdasarkan*`number`*.

Akses atribut

`WHERE`Klausul `SELECT` dan dapat merujuk ke data catatan dengan menggunakan salah satu metode di bagian-bagian berikut, tergantung pada apakah file yang dicari dalam format CSV atau JSON.

CSV

- Nomor Kolom — Anda dapat merujuk ke kolom Nth pada baris dengan nama kolom_*N*, dengan posisi kolom. *N* Hitungan posisi dimulai pada 1. Misalnya, kolom pertama diberi nama _1 dan kolom kedua diberi nama _2.

  Anda dapat merujuk ke kolom sebagai _*N* atau *`alias`*`.`_*N*. Misalnya, _2 dan `myAlias.`_2 adalah cara yang valid untuk merujuk ke kolom dalam daftar `SELECT` dan klausa `WHERE`.

- Header Kolom – Untuk objek dalam format CSV yang memiliki baris header, header tersedia untuk daftar `SELECT` dan klausul `WHERE`. Secara khusus, seperti dalam SQL tradisional, di dalam ekspresi klausul `SELECT` dan `WHERE`, Anda dapat melihat kolom dengan *`alias`*`.`*`column_name`* atau *`column_name`*.

JSON

- Dokumen— Anda dapat mengakses bidang dokumen JSON sebagai *`alias`*`.`*`name`*. Anda juga dapat mengakses bidang bersarang, misalnya,*`alias`*`.`*`name1`*`.`*`name2`*`.`*`name3`*.
- Daftar — Anda dapat mengakses elemen dalam daftar JSON dengan menggunakan indeks berbasis nol dengan operator. `[]` Misalnya, Anda dapat mengakses elemen kedua dari daftar sebagai *`alias`*`[1]`. Anda dapat menggabungkan mengakses elemen daftar dengan bidang, misalnya,*`alias`*`.`*`name1`*`.`*`name2`*`[1].`*`name3`*.
- Contoh: Pertimbangkan objek JSON ini sebagai sampel set data:

```
{"name": "Susan Smith",
"org": "engineering",
"projects":
    [
     {"project_name":"project1", "completed":false},
     {"project_name":"project2", "completed":true}
    ]
}
```

Contoh #1

Kueri berikut mengembalikan hasil ini:

```
Select s.name from S3Object s
```

```
{"name":"Susan Smith"}
```

Contoh #2

Kueri berikut mengembalikan hasil ini:

```
Select s.projects[0].project_name from S3Object s
```

```
{"project_name":"project1"}
```

Sensitivitas kasus nama header dan atribut

Dengan Amazon S3 Select, Anda dapat menggunakan tanda petik ganda untuk menunjukkan header kolom (untuk objek CSV) dan atribut (untuk objek CSV) yang peka terhadap huruf besar atau kecil. Tanpa tanda petik ganda, header dan atribut objek tidak sensitif terhadap huruf besar atau kecil. Kesalahan akan muncul dalam kasus ambiguitas.

Contoh-contoh berikut adalah 1) objek Amazon S3 dalam format CSV dengan header bidang yang ditentukan, dan dengan `FileHeaderInfo` diatur ke `"Use"` untuk permintaan kueri; atau 2) objek Amazon S3 dalam format JSON dengan atribut yang ditentukan.

Contoh #1: Objek yang dikueri memiliki header atau atribut. `NAME`

- Ekspresi berikut berhasil mengembalikan nilai dari objek. Karena tidak ada tanda kutip, kueri tidak peka huruf.

```
SELECT s.name from S3Object s
```

- Ekspresi berikut menghasilkan 400 kesalahan`MissingHeaderName`. Karena ada tanda kutip, query adalah case sensitive.

```
SELECT s."name" from S3Object s
```

Contoh #2: Objek Amazon S3 yang dikueri memiliki satu header atau atribut dengan `NAME` header atau atribut lain dengan. `name`

- Ekspresi berikut menghasilkan 400 kesalahan`AmbiguousFieldName`. Karena tidak ada tanda petik, kueri tidak sensitif terhadap huruf besar atau kecil, jadi kesalahan tersebut dilemparkan.

```
SELECT s.name from S3Object s
```

- Ekspresi berikut berhasil mengembalikan nilai dari objek. Karena ada tanda kutip, kueri peka huruf besar, jadi tidak ada ambiguitas.

```
SELECT s."NAME" from S3Object s
```

Menggunakan kata kunci yang dipesan sebagai istilah yang ditentukan pengguna

Amazon S3 Select memiliki serangkaian kata kunci cadangan yang diperlukan untuk menjalankan ekspresi SQL yang digunakan untuk melakukan kueri konten objek. Kata kunci yang disimpan meliputi nama fungsi, tipe data, operator, dan sebagainya. Dalam beberapa kasus, istilah yang ditentukan pengguna, seperti header kolom (untuk file CSV) atau atribut (untuk objek JSON), dapat berbenturan dengan kata kunci cadangan. Saat ini terjadi, Anda harus menggunakan tanda petik ganda untuk menunjukkan bahwa Anda sengaja menggunakan istilah khusus pengguna yang berbenturan dengan kata kunci yang dicadangkan. Jika tidak, kesalahan 400 parse akan terjadi.

Untuk daftar lengkap kata kunci yang dicadangkan, lihatKata kunci yang dicadangkan.

Contoh berikut adalah 1) objek Amazon S3 dalam format CSV dengan header bidang yang ditentukan, dengan `FileHeaderInfo` diatur ke `"Use"` untuk permintaan kueri, atau 2) objek Amazon S3 dalam format JSON dengan atribut yang ditentukan.

Contoh: Objek yang dikueri memiliki header atau atribut bernama`CAST`, yang merupakan kata kunci cadangan.

- Ekspresi berikut berhasil mengembalikan nilai dari objek. Karena tanda kutip digunakan dalam kueri, S3 Select menggunakan header atau atribut yang ditentukan pengguna.

  ```
  SELECT s."CAST" from S3Object s
  ```

- Ekspresi berikut menghasilkan 400 parse. Karena tidak ada tanda kutip yang digunakan dalam kueri, `CAST` bentrokan dengan kata kunci yang dipesan.

  ```
  SELECT s.CAST from S3Object s
  ```

Ekspresi skalar

Di dalam klausul `WHERE` dan daftar `SELECT`, Anda dapat memiliki ekspresi skalar SQL, yaitu ekspresi yang mengembalikan nilai skalar. Memiliki bentuk sebagai berikut:

- ***literal***

  SQL literal.

- ***column_reference***

  Sebuah referensi ke kolom dalam bentuk *`column_name`* atau*`alias`*`.`*`column_name`*.

- ***unary_op expression***

  Dalam hal ini, ***unary_op***adalah operator SQL unary.

- ***expression binary_op expression***

  Dalam hal ini, ***binary_op***adalah operator biner SQL.

- ***func_name***

  Dalam hal ini, ***func_name***adalah nama fungsi skalar yang diminta.

- ***expression*** `[ NOT ] BETWEEN` ***expression*** `AND` ***expression***
- ***expression*** `LIKE` ***expression*** `[ ESCAPE` ***expression*** `]`

Jenis Data

Amazon S3 Select mendukung beberapa tipe data primitif.

Konversi tipe data

Aturan umumnya adalah mengikuti `CAST` fungsi jika ditentukan. Jika `CAST` tidak ditentukan, kemudian semua data input diperlakukan sebagai string. Dalam hal ini, Anda harus memasukkan data input Anda ke dalam tipe data yang relevan bila perlu.

Untuk informasi lebih lanjut tentang fungsi `CAST`, lihat CAST.

Tipe data yang didukung

Amazon S3 Select mendukung serangkaian tipe data primitif berikut.

Nama	Deskripsi	Contoh
`bool`	Sebuah nilai Boolean, baik `TRUE` atau`FALSE`.	`FALSE`
`int`, `integer`	8-byte integer yang ditandai dengan rentang -9.223.372.036.853.372.036.853.372.036.853.372.036.853.372.036.853.372.036.853.372.036.853.372.036.853.372.036.853.372.036.853.372.036.	`100000`

Nama	Deskripsi	Contoh
`string`	String panjang variabel UTF8-dikodekan. Batas default adalah 1 karakter. Batas karakter maksimum adalah 2.147.483.647.	`'xyz'`
`float`	Nomor 8-byte floating point.	`CAST(0.456 AS FLOAT)`
`decimal`, `numeric`	Angka Dasar-10, dengan presisi maksimum 38 (yaitu, jumlah maksimal digit signifikan), dan dengan skala dalam rentang -2 31 hingga 2 31 hingga 2 31 hingga

Nama	Deskripsi	Contoh
`timestamp`	Timestamp mewakili momen waktu tertentu, selalu menyertakan offset lokal, dan mampu memiliki presisi arbitrer. Dalam format teks, timestamp mengikuti Catatan W3C tentang format tanggal dan waktu, tetapi harus mengakhir inya dengan literal `T` jika timestamp tidak setidaknya presisi sehari penuh. Beberapa detik fraksional diperbolehkan, dengan setidaknya satu digit presisi, dan maksimum tak terbatas. Offset waktu lokal dapat dinyatakan sebagai offset jam:menit dari UTC, atau sebagai literal `Z` untuk menunjukk an waktu lokal UTC. Offset waktu lokal diperlukan pada timestamp dengan waktu dan tidak diizinkan pada nilai tanggal.	`CAST('2007-04-05T14:30Z' AS TIMESTAMP)`

ParquetJenis yang didukung

Amazon S3 Select mendukung Parquet tipe berikut.

- `DATE`
- `DECIMAL`
- `ENUM`
- `INT(8)`
- `INT(16)`
- `INT(32)`
- `INT(64)`
- `LIST`

> **Note**
>
> Untuk keluaran `LIST` Parquet tipe, Amazon S3 Select hanya mendukung format JSON. Namun, jika kueri membatasi data ke nilai sederhana, `LIST` Parquet jenisnya juga dapat ditanyakan dalam format CSV.

- `STRING`
- `TIMESTAMP`didukung presisi (`MILLIS`/`MICROS`/`NANOS`)

> **Note**
>
> Stempel waktu yang disimpan sebagai tidak `INT(96)` didukung.
> Karena kisaran `INT(64)` jenis, cap waktu yang menggunakan `NANOS` unit dapat mewakili hanya nilai-nilai antara `1677-09-21 00:12:43` dan. `2262-04-11 23:47:16` Nilai di luar kisaran ini tidak dapat direpresentasikan dengan `NANOS` unit.

Pemetaan jenis ke Parquet tipe data yang didukung di Amazon S3 Select

Parquetjenis	Tipe data yang didukung
`DATE`	`timestamp`
`DECIMAL`	`decimal`, `numeric`
`ENUM`	`string`
`INT(8)`	`int`, `integer`
`INT(16)`	`int`, `integer`
`INT(32)`	`int`, `integer`
`INT(64)`	`decimal`, `numeric`

Parquetjenis	Tipe data yang didukung
`LIST`	Setiap Parquet jenis dalam daftar dipetakan ke tipe data yang sesuai.
`STRING`	`string`
`TIMESTAMP`	`timestamp`

Operator

Amazon S3 Select mendukung operator berikut.

Operator logistik

- `AND`
- `NOT`
- `OR`

Operator perbandingan

- `<`
- `>`
- `<=`
- `>=`
- `=`
- `<>`
- `!=`
- `BETWEEN`
- `IN` – Misalnya: `IN ('a', 'b', 'c')`

Operator pencocokan pola

- `LIKE`

- _ (Cocok karakter apa pun)
- % (Cocok urutan karakter apa pun)

Operator kesatuan

- `IS NULL`
- `IS NOT NULL`

Operator Matematika

Penambahan, pengurangan, perkalian, pembagian, dan modulo didukung, sebagai berikut:

- `+`
- `-`
- `*`
- `/`
- `%`

Operator diutamakan

Tabel berikut menunjukkan prioritas operator dalam urutan menurun.

Operator atau elemen	Asosiatif	Wajib
`-`	kanan	unary minus
`*`, `/`, `%`	kiri	perkalian, pembagian, modulo
`+`, `-`	kiri	tambahan, pengurangan
`IN`		atur keanggotaan

Operator atau elemen	Asosiatif	Wajib
BETWEEN		pengurungan rentang
LIKE		pencocokan pola string
<>		lebih kecil dari, lebih besar dari
=	kanan	kesetaraan, penugasan
NOT	kanan	negasi logis
AND	kiri	konjungsi logis
OR	kiri	disjungsi logis

Kata kunci yang dicadangkan

Berikut ini adalah daftar kata kunci cadangan untuk Amazon S3 Select. Kata kunci ini mencakup nama fungsi, tipe data, operator, dan sebagainya, yang diperlukan untuk menjalankan ekspresi SQL yang digunakan untuk mencari konten objek.

```
absolute
action
add
all
allocate
alter
and
any
are
as
asc
assertion
at
```

```
authorization
avg
bag
begin
between
bit
bit_length
blob
bool
boolean
both
by
cascade
cascaded
case
cast
catalog
char
char_length
character
character_length
check
clob
close
coalesce
collate
collation
column
commit
connect
connection
constraint
constraints
continue
convert
corresponding
count
create
cross
current
current_date
current_time
current_timestamp
current_user
```

```
cursor
date
day
deallocate
dec
decimal
declare
default
deferrable
deferred
delete
desc
describe
descriptor
diagnostics
disconnect
distinct
domain
double
drop
else
end
end-exec
escape
except
exception
exec
execute
exists
external
extract
false
fetch
first
float
for
foreign
found
from
full
get
global
go
goto
```

```
grant
group
having
hour
identity
immediate
in
indicator
initially
inner
input
insensitive
insert
int
integer
intersect
interval
into
is
isolation
join
key
language
last
leading
left
level
like
limit
list
local
lower
match
max
min
minute
missing
module
month
names
national
natural
nchar
next
```

```
no
not
null
nullif
numeric
octet_length
of
on
only
open
option
or
order
outer
output
overlaps
pad
partial
pivot
position
precision
prepare
preserve
primary
prior
privileges
procedure
public
read
real
references
relative
restrict
revoke
right
rollback
rows
schema
scroll
second
section
select
session
session_user
```

```
set
sexp
size
smallint
some
space
sql
sqlcode
sqlerror
sqlstate
string
struct
substring
sum
symbol
system_user
table
temporary
then
time
timestamp
timezone_hour
timezone_minute
to
trailing
transaction
translate
translation
trim
true
tuple
union
unique
unknown
unpivot
update
upper
usage
user
using
value
values
varchar
varying
```

```
view
when
whenever
where
with
work
write
year
zone
```

Fungsi SQL

Amazon S3 Select mendukung fungsi SQL berikut.

Topik

- Fungsi agregat
- Fungsi kondisional
- Fungsi konversi
- Fungsi tanggal
- Fungsi string

Fungsi agregat

Amazon S3 Select mendukung fungsi agregat berikut.

Fungsi	Jenis argumen	Jenis pengembalian
`AVG(`*`expression`*`)`	`INT`, `FLOAT`, `DECIMAL`	`DECIMAL`untuk `INT` argumen, `FLOAT` untuk argumen titik mengambang; jika tidak, sama dengan tipe data argumen.
`COUNT`	-	`INT`

Fungsi	Jenis argumen	Jenis pengembalian
`MAX(`*expression*`)`	`INT`, `DECIMAL`	Sama seperti tipe argumen.
`MIN(`*expression*`)`	`INT`, `DECIMAL`	Sama seperti tipe argumen.
`SUM(`*expression*`)`	`INT`, `FLOAT`, `DOUBLE`, `DECIMAL`	`INT`untuk `INT` argumen, `FLOAT` untuk argumen titik mengambang; jika tidak, sama dengan tipe data argumen.

Contoh SUM

Untuk menggabungkan ukuran objek total folder dalam laporan S3 Inventory, gunakan ekspresi. `SUM`

Laporan S3 Inventory berikut adalah file CSV yang dikompresi dengan GZIP. Ada tiga kolom.

- Kolom pertama adalah nama bucket S3 (*`DOC-EXAMPLE-BUCKET`*) yang digunakan untuk laporan S3 Inventory.
- Kolom kedua adalah nama kunci objek yang secara unik mengidentifikasi objek tersebut dalam bucket.

  *`example-folder/`*Nilai di baris pertama adalah untuk folder*`example-folder`*. Di Amazon S3, saat Anda membuat folder di bucket, S3 membuat objek 0-byte dengan kunci yang disetel ke nama folder yang Anda berikan.

  *`example-folder/object1`*Nilai di baris kedua adalah untuk objek *`object1`* dalam folder*`example-folder`*.

  *`example-folder/object2`*Nilai di baris ketiga adalah untuk objek *`object2`* dalam folder*`example-folder`*.

Untuk informasi selengkapnya tentang folder S3, lihatMengatur objek di konsol Amazon S3 dengan menggunakan folder.

- Kolom ketiga adalah ukuran objek dalam byte.

```
"DOC-EXAMPLE-BUCKET","example-folder/","0"
"DOC-EXAMPLE-BUCKET","example-folder/object1","2011267"
"DOC-EXAMPLE-BUCKET","example-folder/object2","1570024"
```

Untuk menggunakan `SUM` ekspresi untuk menghitung ukuran total folder*`example-folder`*, jalankan kueri SQL dengan Amazon S3 Select.

```
SELECT SUM(CAST(_3 as INT)) FROM s3object s WHERE _2 LIKE 'example-folder/%' AND _2 !=
 'example-folder/';
```

Hasil Kueri:

```
3581291
```

Fungsi kondisional

Amazon S3 Select mendukung fungsi kondisional berikut.

Topik

- CASE
- COALESCE
- NULLIF

CASE

`CASE`Ekspresi tersebut adalah ekspresi bersyarat, mirip dengan `if/then/else` pernyataan yang ditemukan dalam bahasa lain. `CASE`digunakan untuk menentukan hasil jika terdapat beberapa kondisi. Ada dua jenis `CASE` ekspresi: sederhana dan dicari.

Dalam `CASE` ekspresi sederhana, ekspresi dibandingkan dengan nilai. Ketika kecocokan ditemukan, tindakan tertentu dalam `THEN` klausul diterapkan. Jika tidak ada kecocokan ditemukan, tindakan dalam `ELSE` klausul diterapkan.

Dalam `CASE` ekspresi yang dicari, masing-masing `CASE` dievaluasi berdasarkan ekspresi Boolean, dan `CASE` pernyataan mengembalikan pencocokan pertama. `CASE` Jika tidak ada kecocokan `CASE` ditemukan di antara `WHEN` klausul, tindakan dalam `ELSE` klausul dikembalikan.

Sintaksis

> **Note**
>
> Saat ini, Amazon S3 Select tidak mendukung `ORDER BY` atau kueri yang berisi baris baru. Pastikan Anda menggunakan kueri tanpa jeda baris.

Berikut ini adalah `CASE` pernyataan sederhana yang digunakan untuk mencocokkan kondisi:

```
CASE expression WHEN value THEN result [WHEN...] [ELSE result] END
```

Berikut ini adalah `CASE` pernyataan yang dicari yang digunakan untuk mengevaluasi setiap kondisi:

```
CASE WHEN boolean condition THEN result [WHEN ...] [ELSE result] END
```

Contoh

> **Note**
>
> Jika Anda menggunakan konsol Amazon S3 untuk menjalankan contoh berikut dan file CSV Anda berisi baris header, pilih Kecualikan baris pertama data CSV.

Contoh 1: Gunakan `CASE` ekspresi sederhana untuk mengganti `New York City` dengan `Big Apple` dalam query. Ganti semua nama kota lainnya dengan`other`.

```
SELECT venuecity, CASE venuecity WHEN 'New York City' THEN 'Big Apple' ELSE 'other' END
 FROM S3Object;
```

Hasil Kueri:

```
venuecity        |   case
-----------------+-----------
Los Angeles      | other
```

```
New York City    | Big Apple
San Francisco    | other
Baltimore        | other
...
```

Contoh 2: Gunakan `CASE` ekspresi yang dicari untuk menetapkan nomor grup berdasarkan `pricepaid` nilai penjualan tiket individu:

```
SELECT pricepaid, CASE WHEN CAST(pricepaid as FLOAT) < 10000 THEN 'group 1' WHEN
 CAST(pricepaid as FLOAT) > 10000 THEN 'group 2' ELSE 'group 3' END FROM S3Object;
```

Hasil Kueri:

```
pricepaid |  case
-----------+---------
12624.00 | group 2
10000.00 | group 3
10000.00 | group 3
9996.00 | group 1
9988.00 | group 1
...
```

COALESCE

`COALESCE`mengevaluasi argumen secara berurutan dan mengembalikan nilai non-diketahui pertama, yaitu, nilai non-null atau non-hilang yang pertama. Fungsi ini tidak memperbanyak nilai null dan hilang.

Sintaks

```
COALESCE ( expression, expression, ... )
```

Parameter

`expression`

Ekspresi target yang fungsinya berjalan.

Contoh

```
COALESCE(1)                -- 1
```

```
COALESCE(null)             -- null
COALESCE(null, null)       -- null
COALESCE(missing)          -- null
COALESCE(missing, missing) -- null
COALESCE(1, null)          -- 1
COALESCE(null, null, 1)    -- 1
COALESCE(null, 'string')   -- 'string'
COALESCE(missing, 1)       -- 1
```

NULLIF

Dengan dua ekspresi, `NULLIF` kembalikan `NULL` jika kedua ungkapan tersebut mengevaluasi nilai yang sama; jika tidak, `NULLIF` kembalikan hasil evaluasi ke ekspresi yang pertama.

Sintaks

```
NULLIF ( expression1, expression2 )
```

Parameter

`expression1`, *`expression2`*

Ekspresi target yang menjalankan fungsi.

Contoh

```
NULLIF(1, 1)             -- null
NULLIF(1, 2)             -- 1
NULLIF(1.0, 1)           -- null
NULLIF(1, '1')           -- 1
NULLIF([1], [1])         -- null
NULLIF(1, NULL)          -- 1
NULLIF(NULL, 1)          -- null
NULLIF(null, null)       -- null
NULLIF(missing, null)    -- null
NULLIF(missing, missing) -- null
```

Fungsi konversi

Amazon S3 Select mendukung fungsi konversi berikut.

Topik

- CAST

CAST

Fungsi CAST mengonversi suatu entitas, seperti ekspresi yang mengevaluasi ke nilai tunggal, dari satu tipe ke lainnya.

Sintaks

```
CAST ( expression AS data_type )
```

Parameter

expression

Kombinasi satu nilai atau lebih, operator, dan fungsi SQL yang mengevaluasi nilai.

data_type

Tipe data target, seperti INT, untuk menghadirkan ekspresi. Untuk daftar tipe data yang didukung, lihat Jenis Data.

Contoh

```
CAST('2007-04-05T14:30Z' AS TIMESTAMP)
CAST(0.456 AS FLOAT)
```

Fungsi tanggal

Amazon S3 Select mendukung fungsi tanggal berikut.

Topik

- DATE_ADD
- DATE_DIFF
- EXTRACT
- TO_STRING
- TO_TIMESTAMP
- UTCNOW

DATE_ADD

Dengan bagian tanggal, kuantitas, dan timestamp, `DATE_ADD` mengembalikan timestamp yang diperbarui dengan mengubah bagian tanggal berdasarkan kuantitas.

Sintaks

```
DATE_ADD( date_part, quantity, timestamp )
```

Parameter

`date_part`

Menentukan bagian mana dari tanggal untuk diubah. Ini dapat menjadi salah satu dari yang berikut:

- tahun
- bulan
- hari
- jam
- menit
- detik

`quantity`

Nilai yang diterapkan pada stempel waktu yang diperbarui. Nilai positif untuk *`quantity`* ditambahkan ke date_part timestamp, dan pengurangan nilai negatif.

`timestamp`

Stempel waktu target yang menjalankan fungsi.

Contoh

```
DATE_ADD(year, 5, `2010-01-01T`)                -- 2015-01-01 (equivalent to
 2015-01-01T)
DATE_ADD(month, 1, `2010T`)                     -- 2010-02T (result will add precision
 as necessary)
DATE_ADD(month, 13, `2010T`)                    -- 2011-02T
DATE_ADD(day, -1, `2017-01-10T`)                -- 2017-01-09 (equivalent to
 2017-01-09T)
```

```
DATE_ADD(hour, 1, `2017T`)                      -- 2017-01-01T01:00-00:00
DATE_ADD(hour, 1, `2017-01-02T03:04Z`)          -- 2017-01-02T04:04Z
DATE_ADD(minute, 1, `2017-01-02T03:04:05.006Z`) -- 2017-01-02T03:05:05.006Z
DATE_ADD(second, 1, `2017-01-02T03:04:05.006Z`) -- 2017-01-02T03:04:06.006Z
```

DATE_DIFF

Dengan bagian tanggal dan dua stempel waktu yang valid, `DATE_DIFF` mengembalikan selisih bagian tanggal. Nilai kembalian adalah bilangan bulat negatif ketika nilai *`date_part`* dari *`timestamp1`* lebih besar dari nilai *`date_part`* dari *`timestamp2`*. Nilai kembalian adalah bilangan bulat positif ketika nilai *`date_part`* dari *`timestamp1`* lebih kecil dari nilai *`date_part`* dari *`timestamp2`*.

Sintaks

```
DATE_DIFF( date_part, timestamp1, timestamp2 )
```

Parameter

`date_part`

Menentukan bagian mana dari stempel waktu untuk dibandingkan. Untuk definisi `date_part`, lihat DATE_ADD.

`timestamp1`

Stempel waktu pertama untuk dibandingkan.

`timestamp2`

Stempel waktu kedua untuk dibandingkan.

Contoh

```
DATE_DIFF(year, `2010-01-01T`, `2011-01-01T`)            -- 1
DATE_DIFF(year, `2010T`, `2010-05T`)                     -- 4 (2010T is equivalent to
 2010-01-01T00:00:00.000Z)
DATE_DIFF(month, `2010T`, `2011T`)                       -- 12
DATE_DIFF(month, `2011T`, `2010T`)                       -- -12
DATE_DIFF(day, `2010-01-01T23:00`, `2010-01-02T01:00`) -- 0 (need to be at least 24h
 apart to be 1 day apart)
```

EXTRACT

Dengan bagian tanggal dan timestamp, `EXTRACT` mengembalikan nilai bagian tanggal timestamp.

Sintaks

```
EXTRACT( date_part FROM timestamp )
```

Parameter

`date_part`

Menentukan bagian mana dari stempel waktu untuk mengekstrak. Ini dapat berupa salah satu dari berikut ini:

- `YEAR`
- `MONTH`
- `DAY`
- `HOUR`
- `MINUTE`
- `SECOND`
- `TIMEZONE_HOUR`
- `TIMEZONE_MINUTE`

`timestamp`

Stempel waktu target yang menjalankan fungsi.

Contoh

```
EXTRACT(YEAR FROM `2010-01-01T`)                          -- 2010
EXTRACT(MONTH FROM `2010T`)                               -- 1 (equivalent to
 2010-01-01T00:00:00.000Z)
EXTRACT(MONTH FROM `2010-10T`)                            -- 10
EXTRACT(HOUR FROM `2017-01-02T03:04:05+07:08`)            -- 3
EXTRACT(MINUTE FROM `2017-01-02T03:04:05+07:08`)          -- 4
EXTRACT(TIMEZONE_HOUR FROM `2017-01-02T03:04:05+07:08`)   -- 7
EXTRACT(TIMEZONE_MINUTE FROM `2017-01-02T03:04:05+07:08`) -- 8
```

TO_STRING

Dengan timestamp dan pola format, `TO_STRING` mengembalikan representasi string dari timestamp dalam format yang diberikan.

Sintaks

```
TO_STRING ( timestamp time_format_pattern )
```

Parameter

`timestamp`

Stempel waktu target yang menjalankan fungsi.

`time_format_pattern`

String yang memiliki interpretasi karakter khusus berikut:

Format	Contoh	Deskripsi
`yy`	`69`	2 digit tahun
`y`	`1969`	4 digit tahun
`yyyy`	`1969`	4 digit tahun yang ditambahkan nol
`M`	`1`	Bulan tahun
`MM`	`01`	Bulan dalam tahun yang ditambahkan nol
`MMM`	`Jan`	Nama bulan dalam tahun disingkat

Format	Contoh	Deskripsi
`MMMM`	`January`	Nama bulan dalam tahun ditulis penuh
`MMMMM`	`J`	Huruf pertama bulan dalam tahun (CATATAN: Format ini tidak valid untuk digunakan dengan `TO_TIMESTAMP` fungsi.)
d	`2`	Hari dalam bulan (1-31)
dd	`02`	Hari pada bulan yang ditambahkan nol (01-31)
a	`AM`	AM atau PM pada hari
h	`3`	Jam dalam hari (1-12)
hh	`03`	Jam dalam hari yang ditambahkan nol (01-12)

Format	Contoh	Deskripsi
`H`	`3`	Jam dalam hari (0-23)
`HH`	`03`	Jam dalam hari yang ditambahkan nol (00-23)
`m`	`4`	Menit dalam jam (0-59)
`mm`	`04`	Menit dalam jam yang ditambahkan nol (00-59)
`s`	`5`	Detik dalam menit (0-59)
`ss`	`05`	Detik dalam menit yang ditambahkan nol (00-59)
`S`	`0`	Fraksi detik (presisi: 0,1, rentang: 0,0-0,9)
`SS`	`6`	Fraksi detik (presisi: 0,01, rentang: 0,0-0,99)

Format	Contoh	Deskripsi
SSS	60	Fraksi detik (presisi: 0,001, rentang: 0,0-0,999)
…	…	…
SSSSSSSSS	60000000	Fraksi detik (presisi maksimum: 1 nanodetik , rentang: 0,0-0,033 .373.373. 373.373.3 73.373.37 3.373.373 .373.373. 373.373.3 73.373.37 3.379.
n	60000000	Nano dari detik
X	+07 atau Z	Offset dalam jam, atau Z jika offset adalah 0
XX atau XXXX	+0700 atau Z	Offset dalam jam dan menit, atau Z jika offset adalah 0

Format	Contoh	Deskripsi
XXX atau XXXXX	`+07:00` atau Z	Offset dalam jam dan menit, atau Z jika offset adalah 0
x	7	Offset dalam jam
`xx` atau `xxxx`	`700`	Offset dalam jam dan menit
`xxx` atau `xxxxx`	`+07:00`	Offset dalam jam dan menit

Contoh

```
TO_STRING(`1969-07-20T20:18Z`,  'MMMM d, y')                    -- "July 20, 1969"
TO_STRING(`1969-07-20T20:18Z`, 'MMM d, yyyy')                   -- "Jul 20, 1969"
TO_STRING(`1969-07-20T20:18Z`, 'M-d-yy')                        -- "7-20-69"
TO_STRING(`1969-07-20T20:18Z`, 'MM-d-y')                        -- "07-20-1969"
TO_STRING(`1969-07-20T20:18Z`, 'MMMM d, y h:m a')               -- "July 20, 1969 8:18
 PM"
TO_STRING(`1969-07-20T20:18Z`, 'y-MM-dd''T''H:m:ssX')           --
 "1969-07-20T20:18:00Z"
TO_STRING(`1969-07-20T20:18+08:00Z`, 'y-MM-dd''T''H:m:ssX')     --
 "1969-07-20T20:18:00Z"
TO_STRING(`1969-07-20T20:18+08:00`, 'y-MM-dd''T''H:m:ssXXXX')   --
 "1969-07-20T20:18:00+0800"
TO_STRING(`1969-07-20T20:18+08:00`, 'y-MM-dd''T''H:m:ssXXXXX')  --
 "1969-07-20T20:18:00+08:00"
```

TO_TIMESTAMP

Dengan string, `TO_TIMESTAMP` mengonversi menjadi stempel waktu. `TO_TIMESTAMP`adalah operasi terbalik. `TO_STRING`

Sintaks

```
TO_TIMESTAMP ( string )
```

Parameter

string

String target tempat fungsi beroperasi.

Contoh

```
TO_TIMESTAMP('2007T')                          -- `2007T`
TO_TIMESTAMP('2007-02-23T12:14:33.079-08:00') -- `2007-02-23T12:14:33.079-08:00`
```

UTCNOW

`UTCNOW`mengembalikan waktu saat ini dalam UTC sebagai stempel waktu.

Sintaks

```
UTCNOW()
```

Parameter

`UTCNOW`tidak menggunakan parameter.

Contoh

```
UTCNOW() -- 2017-10-13T16:02:11.123Z
```

Fungsi string

Amazon S3 Select mendukung fungsi string berikut.

Topik

- CHAR_LENGTH, CHARACTER_LENGTH
- LOWER
- SUBSTRING
- TRIM

- UPPER

CHAR_LENGTH, CHARACTER_LENGTH

`CHAR_LENGTH`(atau`CHARACTER_LENGTH`) menghitung jumlah karakter dalam string tertentu.

> **Note**
>
> `CHAR_LENGTH` dan `CHARACTER_LENGTH` adalah sinonim.

Sintaks

```
CHAR_LENGTH ( string )
```

Parameter

`string`

String target tempat fungsi beroperasi.

Contoh

```
CHAR_LENGTH('')          -- 0
CHAR_LENGTH('abcdefg')   -- 7
```

LOWER

Dengan string, `LOWER` mengubah semua karakter huruf besar menjadi karakter huruf kecil. Setiap karakter yang tidak dalam huruf besar tetap tidak berubah.

Sintaks

```
LOWER ( string )
```

Parameter

`string`

String target tempat fungsi beroperasi.

Contoh

```
LOWER('AbCdEfG!@#$') -- 'abcdefg!@#$'
```

SUBSTRING

Dengan string, indeks awal, dan sebagai pilihan panjang, `SUBSTRING` mengembalikan substring dari indeks awal hingga ujung string, atau hingga panjang yang disediakan.

> **Note**
>
> Karakter pertama dari string input memiliki posisi indeks 1.
>
> - Jika `start` <1, tanpa panjang yang ditentukan, maka posisi indeks diatur ke 1.
> - Jika `start` <1, dengan panjang yang ditentukan, maka posisi indeks diatur ke`start + length -1`.
> - Jika `start + length -1` <0, maka string kosong dikembalikan.
> - If `start + length -1` > = 0, maka substring dimulai pada posisi indeks 1 dengan panjang `start + length - 1` dikembalikan.

Sintaks

```
SUBSTRING( string FROM start [ FOR length ] )
```

Parameter

`string`

String target tempat fungsi beroperasi.

`start`

Posisi awal string.

`length`

Panjang substring untuk kembali. Jika tidak ada, lanjutkan ke akhir string.

Contoh

```
SUBSTRING("123456789", 0)      -- "123456789"
SUBSTRING("123456789", 1)      -- "123456789"
SUBSTRING("123456789", 2)      -- "23456789"
SUBSTRING("123456789", -4)     -- "123456789"
SUBSTRING("123456789", 0, 999) -- "123456789"
SUBSTRING("123456789", 1, 5)   -- "12345"
```

TRIM

Memangkas karakter di depan atau di belakang dari sebuah string. Karakter default yang harus dihapus adalah spasi (`' '`).

Sintaks

```
TRIM ( [[LEADING | TRAILING | BOTH remove_chars] FROM] string )
```

Parameter

`string`

String target tempat fungsi beroperasi.

`LEADING` | `TRAILING` | `BOTH`

Parameter ini menunjukkan apakah akan memangkas karakter di depan atau di belakang, atau karakter di depan dan di belakang.

`remove_chars`

Set karakter untuk dihapus. *`remove_chars`*dapat berupa string dengan panjang > 1. Fungsi ini mengembalikan string dengan karakter apa pun dari *`remove_chars`* yang ditemukan di awal atau akhir string yang dihapus.

Contoh

```
TRIM('       foobar         ')              -- 'foobar'
TRIM('      \tfoobar\t         ')           -- '\tfoobar\t'
TRIM(LEADING FROM '       foobar         ')  -- 'foobar         '
TRIM(TRAILING FROM '       foobar         ') -- '       foobar'
```

```
TRIM(BOTH FROM '       foobar         ')     -- 'foobar'
TRIM(BOTH '12' FROM '1112211foobar22211122') -- 'foobar'
```

UPPER

Dengan string, UPPER mengubah semua karakter huruf kecil menjadi karakter huruf besar. Setiap karakter yang tidak dalam huruf kecil tetap tidak berubah.

Sintaks

```
UPPER ( string )
```

Parameter

string

String target tempat fungsi beroperasi.

Contoh

```
UPPER('AbCdEfG!@#$') -- 'ABCDEFG!@#$'
```

Melakukan operasi batch berskala besar pada objek Amazon S3

Anda dapat menggunakan Operasi Batch S3 untuk melakukan operasi batch berskala besar pada objek Amazon S3. Operasi Batch S3 dapat melakukan satu operasi untuk sejumlah objek Amazon S3 yang telah Anda tentukan. Satu pekerjaan dapat melakukan operasi tertentu pada miliaran objek yang berisi data berukuran exabyte. Amazon S3 melacak kemajuan, mengirim pemberitahuan, dan menyimpan laporan penyelesaian terperinci dari semua tindakan, menyediakan pengalaman yang sepenuhnya terkelola, dapat diaudit, dan nirserver. Anda dapat menggunakan Operasi Batch S3 melalui AWS Management Console, AWS CLI, SDK Amazon, atau API REST.

Gunakan Operasi Batch S3 untuk menyalin objek dan mengatur tanda objek atau daftar kontrol akses (ACL). Anda juga dapat memulai pemulihan objek dari S3 Glacier Flexible Retrieval atau menginvokasi fungsi AWS Lambda untuk melakukan tindakan khusus menggunakan objek Anda. Anda dapat melakukan operasi ini pada daftar objek kustom, atau menggunakan laporan inventaris Amazon S3 untuk membuat daftar objek dengan mudah. Operasi Batch Amazon S3 menggunakan

API Amazon S3 yang sama dengan yang sudah Anda gunakan dengan Amazon S3, sehingga Anda akan terbiasa dengan antarmukanya.

> **Note**
>
> Untuk informasi selengkapnya tentang penggunaan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori. Untuk informasi selengkapnya tentang penggunaan Operasi Batch dengan S3 Express One Zone dan bucket direktori, lihat Menggunakan Operasi Batch dengan S3 Express One Zone.

Dasar-dasar Operasi Batch S3

Anda dapat menggunakan Operasi Batch S3 untuk melakukan operasi batch berskala besar pada objek Amazon S3. Operasi Batch S3 dapat menjalankan operasi tunggal atau tindakan pada sejumlah objek Amazon S3 yang telah Anda tentukan.

Terminologi

Bagian ini menggunakan istilah pekerjaan, operasi, dan tugas, yang didefinisikan sebagai berikut:

Pekerjaan

Pekerjaan adalah unit kerja dasar untuk Operasi Batch S3. Pekerjaan berisi semua informasi yang diperlukan untuk menjalankan operasi tertentu pada objek yang tercantum dalam manifes. Setelah Anda memberikan informasi ini dan meminta agar pekerjaan tersebut dimulai, pekerjaan akan menjalankan operasi untuk setiap objek dalam manifes.

Operasi

Operasi adalah jenis tindakan API, seperti menyalin objek yang akan dijalankan pekerjaan Operasi Batch. Setiap pekerjaan melakukan satu jenis operasi di semua objek yang telah ditentukan dalam manifes.

Tugas

Tugas adalah unit pelaksanaan pekerjaan. Tugas mewakili panggilan tunggal ke operasi Amazon S3 atau operasi API AWS Lambda untuk melakukan operasi pekerjaan pada satu objek. Selama masa aktif pekerjaan, Operasi Batch S3 membuat satu tugas untuk setiap objek yang ditentukan dalam manifes.

Cara kerja pekerjaan Operasi Batch S3

Pekerjaan adalah unit kerja dasar untuk Operasi Batch S3. Pekerjaan berisi semua informasi yang diperlukan untuk menjalankan operasi tertentu pada sejumlah objek. Untuk membuat pekerjaan, Anda memberikan daftar objek kepada Operasi Batch S3 dan menentukan tindakan yang harus dilakukan pada objek tersebut.

Untuk informasi tentang operasi yang didukung Operasi Batch S3, lihat Operasi yang didukung oleh Operasi Batch S3.

Pekerjaan batch melakukan operasi tertentu pada setiap objek yang disertakan dalam manifes. Manifes mencantumkan objek yang harus diproses pekerjaan batch dan disimpan sebagai objek dalam bucket. Anda dapat menggunakan laporan Inventaris Amazon S3 berformat nilai yang dipishakan koma (CSV) sebagai manifes, yang memudahkan pembuatan daftar besar objek yang berada di bucket. Anda juga dapat menentukan manifes dalam format CSV sederhana yang memungkinkan Anda melakukan operasi batch pada daftar objek yang telah disesuaikan yang dimuat dalam satu bucket.

Setelah Anda membuat pekerjaan, Amazon S3 memproses daftar objek dalam manifes dan menjalankan operasi yang telah ditentukan terhadap setiap objek. Saat pekerjaan berjalan, Anda dapat memantau kemajuannya secara terprogram atau melalui konsol Amazon S3. Anda juga dapat mengonfigurasi pekerjaan untuk menghasilkan laporan penyelesaian setelah selesai. Laporan penyelesaian menguraikan hasil dari setiap tugas yang dilakukan oleh pekerjaan. Untuk informasi selengkapnya tentang cara memantau pekerjaan, lihat Mengelola tugas Operasi Batch S3.

Tutorial Operasi Batch S3

Tutorial berikut menyajikan end-to-end prosedur lengkap untuk beberapa tugas Operasi Batch.

- Tutorial: Video transcoding batch dengan Operasi Batch S3,, dan AWS LambdaAWS Elemental MediaConvert

Memberikan izin untuk Operasi Batch Amazon S3

Sebelum membuat dan menjalankan pekerjaan Operasi Batch S3, Anda harus memberikan izin yang diperlukan. Untuk membuat pekerjaan Operasi Batch Amazon S3, izin pengguna `s3:CreateJob` diperlukan. Entitas yang sama yang membuat pekerjaan juga harus memiliki `iam:PassRole` izin untuk meneruskan peran AWS Identity and Access Management (IAM) yang ditentukan untuk pekerjaan tersebut ke Operasi Batch.

Untuk informasi umum tentang menentukan sumber daya IAM, lihat Kebijakan JSON IAM, Elemen sumber daya di Panduan Pengguna IAM. Bagian berikut memberikan informasi tentang membuat peran IAM dan melampirkan kebijakan.

Topik

- Membuat peran IAM Operasi Batch S3
- Melampirkan kebijakan izin

Membuat peran IAM Operasi Batch S3

Amazon S3 harus memiliki izin untuk menjalankan Operasi Batch S3 atas nama Anda. Anda memberikan izin ini melalui peran AWS Identity and Access Management (IAM). Bagian ini memberikan contoh kebijakan kepercayaan dan izin yang Anda gunakan saat membuat peran IAM. Untuk informasi selengkapnya, lihat peran IAM dalam Panduan Pengguna IAM. Sebagai contoh, lihat Mengontrol izin untuk Operasi Batch S3 menggunakan tanda tugas dan Menyalin objek menggunakan Operasi Batch S3.

Dalam kebijakan IAM, Anda juga dapat menggunakan kunci persyaratan untuk memfilter izin akses yang akan digunakan untuk pekerjaan Operasi Batch S3. Untuk informasi selengkapnya dan daftar lengkap kunci kondisi khusus Amazon S3, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Kebijakan kepercayaan

Untuk mengizinkan pengguna utama layanan Operasi Batch S3 agar dapat menjalankan peran IAM, lampirkan kebijakan kepercayaan berikut ke peran tersebut.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"batchoperations.s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

Melampirkan kebijakan izin

Tergantung pada jenis operasinya, Anda dapat melampirkan salah satu kebijakan berikut.

Sebelum Anda mengonfigurasi izin, perhatikan hal berikut ini:

- Apa pun operasinya, Amazon S3 tetap memerlukan izin untuk membaca objek manifes Anda dari bucket S3 dan menulis laporan ke bucket Anda secara opsional. Oleh karena itu, semua kebijakan berikut mencakup izin ini.
- Untuk manifes laporan inventaris Amazon S3, Operasi Batch S3 memerlukan izin untuk membaca objek manifest.json dan semua file data CSV terkait.
- Izin khusus versi seperti `s3:GetObjectVersion` hanya diperlukan saat Anda menentukan ID versi objek.
- Jika Anda menjalankan Operasi Batch S3 pada objek terenkripsi, peran IAM juga harus memiliki akses ke AWS KMS kunci yang digunakan untuk mengenkripsi mereka.
- Jika Anda mengirimkan manifes laporan inventaris yang dienkripsi AWS KMS, kebijakan IAM Anda harus menyertakan izin `"kms:Decrypt"` dan `"kms:GenerateDataKey"` untuk objek manifest.json serta semua file data CSV terkait.
- Jika pekerjaan Operasi Batch menghasilkan manifes dalam bucket yang mengaktifkan ACL dan berada di AWS akun lain, Anda harus memberikan `s3:PutObjectAcl` izin dalam kebijakan IAM untuk peran IAM yang dikonfigurasi untuk pekerjaan batch. Jika Anda tidak menyertakan izin ini, pekerjaan batch gagal dengan kesalahan`Error occurred when preparing manifest: Failed to write manifest`.

Salin objek: PutObject

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:PutObjectTagging"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:s3:::DestinationBucket/*"
        },
```

```
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:GetObjectTagging",
                "s3:ListBucket"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::SourceBucket",
                "arn:aws:s3:::SourceBucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::ManifestBucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::ReportBucket/*"
            ]
        }
    ]
}
```

Ganti penandaan objek: PutObjectTagging

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
```

```
        "s3:PutObjectTagging",
        "s3:PutObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::TargetResource/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": [
        "arn:aws:s3:::ManifestBucket/*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObject"
      ],
      "Resource":[
        "arn:aws:s3:::ReportBucket/*"
      ]
    }
  ]
}
```

Hapus penandaan objek: DeleteObjectTagging

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "s3:DeleteObjectTagging",
              "s3:DeleteObjectVersionTagging"
            ],
            "Resource": [
                "arn:aws:s3:::TargetResource/*"
            ]
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::ManifestBucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::ReportBucket/*"
            ]
        }
    ]
}
```

Ganti daftar kontrol akses: PutObjectAcl

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObjectAcl",
        "s3:PutObjectVersionAcl"
      ],
      "Resource": "arn:aws:s3:::TargetResource/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": [
        "arn:aws:s3:::ManifestBucket/*"
      ]
```

```
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObject"
      ],
      "Resource":[
        "arn:aws:s3:::ReportBucket/*"
      ]
    }
  ]
}
```

Kembalikan objek: RestoreObject

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
          "s3:RestoreObject"
      ],
      "Resource": "arn:aws:s3:::TargetResource/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": [
        "arn:aws:s3:::ManifestBucket/*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObject"
      ],
      "Resource":[
        "arn:aws:s3:::ReportBucket/*"
      ]
```

```
    }
  ]
}
```

Terapkan retensi Object Lock: PutObjectRetention

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetBucketObjectLockConfiguration",
            "Resource": [
                "arn:aws:s3:::TargetResource"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObjectRetention",
                "s3:BypassGovernanceRetention"
            ],
            "Resource": [
                "arn:aws:s3:::TargetResource/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::ManifestBucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::ReportBucket/*"
```

```
            ]
        }
    ]
}
```

Terapkan penahanan hukum Object Lock: PutObjectLegalHold

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetBucketObjectLockConfiguration",
            "Resource": [
                "arn:aws:s3:::TargetResource"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "s3:PutObjectLegalHold",
            "Resource": [
                "arn:aws:s3:::TargetResource/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::ManifestBucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::ReportBucket/*"
            ]
        }
```

```
    ]
}
```

Replikasi objek yang ada: InitiateReplication dengan manifes yang dihasilkan S3

Gunakan kebijakan ini jika menggunakan dan menyimpan manifes yang dihasilkan S3. Untuk informasi selengkapnya tentang menggunakan Operasi Batch untuk mereplikasi objek yang ada, lihat Mereplikasi objek yang ada dengan Replikasi Batch S3.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Action":[
            "s3:InitiateReplication"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** replication source bucket ***/*"
         ]
      },
      {
         "Action":[
            "s3:GetReplicationConfiguration",
            "s3:PutInventoryConfiguration"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** replication source bucket ***"
         ]
      },
      {
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** manifest bucket ***/*"
         ]
      },
      {
         "Effect":"Allow",
```

```
            "Action":[
               "s3:PutObject"
            ],
            "Resource":[
               "arn:aws:s3:::*** completion report bucket ****/*",
               "arn:aws:s3:::*** manifest bucket ****/*"
            ]
         }
      ]
   }
```

Replikasi objek yang ada: InitiateReplication dengan manifes pengguna

Gunakan kebijakan ini jika Anda menggunakan manifes yang disediakan pengguna. Untuk informasi selengkapnya tentang menggunakan Operasi Batch untuk mereplikasi objek yang ada, lihat Mereplikasi objek yang ada dengan Replikasi Batch S3.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Action":[
            "s3:InitiateReplication"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** replication source bucket ***/*"
         ]
      },
            {
         "Action":[
            "s3:GetObject",
            "s3:GetObjectVersion"
         ],
         "Effect":"Allow",
         "Resource":[
            "arn:aws:s3:::*** manifest bucket ***/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject"
```

```
            ],
            "Resource":[
               "arn:aws:s3:::*** completion report bucket ****/*"
            ]
         }
      ]
}
```

Membuat pekerjaan Operasi Batch S3

Dengan Operasi Batch Amazon S3, Anda dapat melakukan operasi batch berskala besar pada daftar objek Amazon S3 tertentu. Bagian ini menjelaskan informasi yang Anda perlukan untuk membuat pekerjaan Operasi Batch S3 dan hasil permintaan `CreateJob`. Ini juga menyediakan instruksi untuk membuat pekerjaan Operasi Batch dengan menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), dan. AWS SDK for Java

Saat membuat pekerjaan Operasi Batch S3, Anda dapat meminta laporan penyelesaian untuk semua tugas atau hanya untuk tugas yang gagal. Selama setidaknya ada satu tugas yang berhasil diinvokasi, Operasi Batch S3 akan menghasilkan laporan untuk pekerjaan yang telah selesai, gagal, atau dibatalkan. Untuk informasi selengkapnya, lihat Contoh: Laporan penyelesaian Operasi Batch S3.

Topik

- Elemen permintaan pekerjaan Operasi Batch
- Menentukan manifes

Elemen permintaan pekerjaan Operasi Batch

Untuk membuat pekerjaan Operasi Batch S3, Anda harus memberikan informasi berikut:

Operasi

Tentukan operasi yang akan dijalankan Operasi Batch S3 terhadap objek dalam manifes. Setiap jenis operasi menerima parameter khusus untuk operasi tersebut. Dengan Operasi Batch, Anda dapat melakukan operasi secara massal, dengan hasil yang sama seperti jika Anda melakukan operasi itu one-by-one pada setiap objek.

Manifes

Manifes adalah daftar semua objek yang harus dijalankan Operasi Batch S3. Anda dapat menggunakan metode berikut untuk menentukan manifes yang akan digunakan dalam pekerjaan Operasi Batch:

- Buat daftar objek berformat CSV yang disesuaikan secara manual.
- Pilih laporan Inventaris Amazon S3 berformat CSV yang ada.
- Atur Operasi Batch agar menghasilkan manifes secara otomatis berdasarkan kriteria filter objek yang Anda tentukan saat membuat pekerjaan. Opsi ini tersedia untuk pekerjaan replikasi batch yang Anda buat di konsol Amazon S3, atau untuk jenis pekerjaan apa pun yang Anda buat dengan menggunakan AWS CLI AWS , SDK, atau Amazon S3 REST API.

> **Note**
>
> - Terlepas dari cara Anda menentukan manifes, daftar itu harus disimpan di dalam sebuah bucket yang bersifat umum. Operasi Batch tidak dapat mengimpor manifes yang ada dari, atau menyimpan manifes yang dihasilkan ke bucket direktori. Objek yang dijelaskan dalam manifes tetap dapat disimpan dalam direktori bucket. Untuk informasi selengkapnya, lihat Bucket direktori.
> - Jika objek dalam manifes Anda berada dalam bucket yang memiliki versi, menentukan ID versi untuk objek akan mengarahkan Operasi Batch untuk melakukan operasi pada versi tertentu. Jika tidak ada ID versi yang ditentukan, Operasi Batch akan melakukan operasi pada versi terbaru objek. Jika manifes menyertakan bidang ID versi, Anda harus memberikan ID versi untuk semua objek dalam manifes.

Untuk informasi selengkapnya, lihat Menentukan manifes.

Prioritas

Gunakan prioritas pekerjaan untuk menunjukkan prioritas relatif dari pekerjaan ini jika dibandingkan tugas lain yang berjalan di akun Anda. Angka yang lebih tinggi menunjukkan prioritas yang lebih tinggi.

Prioritas pekerjaan hanya memiliki makna relatif terhadap prioritas yang ditetapkan untuk pekerjaan lain di akun dan Wilayah yang sama. Anda bisa memilih sistem penomoran yang sesuai. Misalnya, Anda ingin menetapkan semua pekerjaan Pulihkan (`RestoreObject`) menjadi

prioritas 1, semua pekerjaan Salin (`CopyObject`) menjadi prioritas 2, dan semua pekerjaan Ganti daftar kontrol akses (ACL) (`PutObjectAcl`) menjadi prioritas 3.

Operasi Batch S3 memprioritaskan pekerjaan sesuai dengan urutan prioritas, tetapi tidak menjamin urutan prioritas secara teratur. Oleh karena itu, jangan gunakan prioritas pekerjaan untuk memastikan bahwa satu pekerjaan dimulai atau selesai sebelum pekerjaan lainnya. Jika Anda harus memastikan urutan secara teratur, tunggu hingga satu pekerjaan selesai sebelum memulai pekerjaan berikutnya.

RoleArn

Tentukan peran AWS Identity and Access Management (IAM) untuk menjalankan pekerjaan. Peran IAM yang Anda gunakan harus memiliki izin yang memadai untuk melakukan operasi yang ditentukan dalam pekerjaan. Misalnya, untuk menjalankan pekerjaan `CopyObject`, peran IAM harus memiliki izin `s3:GetObject` untuk bucket sumber dan izin `s3:PutObject` untuk bucket tujuan. Peran ini juga memerlukan izin untuk membaca manifes dan menulis laporan penyelesaian pekerjaan.

Untuk informasi selengkapnya tentang peran IAM, lihat Peran IAM dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang izin Amazon S3, lihat Tindakan kebijakan untuk Amazon S3.

> **Note**
>
> Pekerjaan Operasi Batch yang menjalankan tindakan pada bucket direktori memerlukan izin tertentu. Untuk informasi selengkapnya, lihat AWS Identity and Access Management (IAM) untuk S3 Express One Zone.

Laporan

Tentukan apakah Anda ingin Operasi Batch S3 membuat laporan penyelesaian. Jika meminta laporan penyelesaian pekerjaan, Anda juga harus memberikan parameter untuk laporan dalam elemen ini. Informasi yang diperlukan meliputi:

- Bucket tempat Anda ingin menyimpan laporan

> **Note**
>
> Laporan harus disimpan dalam bucket yang bersifat umum. Operasi Batch tidak dapat menyimpan laporan ke bucket direktori. Untuk informasi selengkapnya, lihat Bucket direktori.

- Format laporan
- Anda ingin laporan menyertakan detail semua tugas atau hanya tugas yang gagal
- String prefiks opsional

> **Note**
>
> Laporan penyelesaian selalu dienkripsi dengan kunci terkelola Amazon S3 (SSE-S3).

Tanda (opsional)

Anda dapat melabeli dan mengontrol akses ke pekerjaan Operasi Batch S3 Anda dengan menambahkan tag. Anda dapat menggunakan tanda untuk mengidentifikasi pihak yang bertanggung jawab atas pekerjaan Operasi Batch, atau untuk mengontrol cara pengguna berinteraksi dengan pekerjaan Operasi Batch. Adanya tanda pekerjaan dapat memberikan atau membatasi kemampuan pengguna untuk membatalkan pekerjaan, mengaktifkan pekerjaan dalam status konfirmasi, atau mengubah tingkat prioritas pekerjaan. Misalnya, Anda dapat memberikan izin pengguna untuk menginvokasi operasi `CreateJob`, asalkan pekerjaan tersebut dibuat dengan tanda `"Department=Finance"`.

Anda dapat membuat pekerjaan dengan tanda yang tersemat, dan dapat menambahkan tanda ke pekerjaan setelah Anda selesai membuatnya.

Untuk informasi selengkapnya, lihat the section called “Menggunakan tanda”.

Deskripsi (opsional)

Untuk melacak dan memantau pekerjaan, Anda juga dapat memberikan deskripsi hingga 256 karakter. Amazon S3 akan menyertakan deskripsi ini setiap memberikan informasi tentang pekerjaan atau menampilkan detail pekerjaan pada konsol Amazon S3. Kemudian, Anda dapat dengan mudah mengurutkan dan memfilter pekerjaan sesuai dengan deskripsi yang Anda tetapkan. Deskripsi tidak harus unik, sehingga Anda dapat menggunakan deskripsi sebagai

kategori (misalnya, "Pekerjaan Menyalin Log Mingguan") untuk memudahkan Anda melacak kelompok pekerjaan yang serupa.

Menentukan manifes

Manifes adalah objek Amazon S3 yang berisi kunci objek yang harus ditindaklanjuti Amazon S3. Anda dapat memberikan manifes dengan salah satu cara berikut:

- Buat file manifes baru secara manual.
- Gunakan manifes yang ada.
- Atur Operasi Batch agar menghasilkan manifes secara otomatis berdasarkan kriteria filter objek yang Anda tentukan saat membuat pekerjaan. Opsi ini tersedia untuk pekerjaan replikasi batch yang Anda buat di konsol Amazon S3, atau untuk jenis pekerjaan apa pun yang Anda buat dengan menggunakan AWS CLI AWS , SDK, atau Amazon S3 REST API.

> **Note**
>
> Terlepas dari cara Anda menentukan manifes, daftar itu harus disimpan di dalam sebuah bucket yang bersifat umum. Operasi Batch tidak dapat mengimpor manifes yang ada dari, atau menyimpan manifes yang dihasilkan ke bucket direktori. Objek yang dijelaskan dalam manifes tetap dapat disimpan dalam direktori bucket. Untuk informasi selengkapnya, lihat Bucket direktori.

Membuat file manifes

Untuk membuat manifes secara manual, Anda perlu menentukan kunci objek manifes, ETag (tanda entitas), dan ID versi opsional dalam daftar yang berformat CSV. Isi manifes harus dienkode URL.

Secara default, Amazon S3 secara otomatis menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) untuk mengenkripsi manifes yang diunggah ke bucket Amazon S3. Manifes yang menggunakan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C) tidak didukung. Manifes yang menggunakan enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS) hanya didukung saat Anda menggunakan laporan inventaris berformat CSV. Menggunakan manifes yang dibuat secara manual dengan AWS KMS tidak didukung.

Manifes Anda harus berisi nama bucket, kunci objek, dan secara opsional, versi objek untuk setiap objek. Bidang lain dalam manifes tidak digunakan oleh Operasi Batch S3.

> **Note**
>
> Jika objek dalam manifes Anda berada dalam bucket yang memiliki versi, menentukan ID versi untuk objek akan mengarahkan Operasi Batch untuk melakukan operasi pada versi tertentu. Jika tidak ada ID versi yang ditentukan, Operasi Batch akan melakukan operasi pada versi terbaru objek. Jika manifes menyertakan bidang ID versi, Anda harus memberikan ID versi untuk semua objek dalam manifes.

Berikut ini adalah contoh manifes dalam format CSV tanpa ID versi.

```
Examplebucket,objectkey1
Examplebucket,objectkey2
Examplebucket,objectkey3
Examplebucket,photos/jpgs/objectkey4
Examplebucket,photos/jpgs/newjersey/objectkey5
Examplebucket,object%20key%20with%20spaces
```

Berikut ini adalah contoh manifes dalam format CSV yang menyertakan ID versi.

```
Examplebucket,objectkey1,PZ9ibn9D5lP6p298B7S9_ceqx1n5EJ0p
Examplebucket,objectkey2,YY_ouuAJByNW1LRBfFMfxMge7XQWxMBF
Examplebucket,objectkey3,jbo9_jhdPEyB4RrmOxWS0kU0EoNrU_oI
Examplebucket,photos/jpgs/objectkey4,6EqlikJJxLTsHsnbZbSRffn24_eh5Ny4
Examplebucket,photos/jpgs/newjersey/objectkey5,imHf3FAiRsvBW_EHB8GOu.NHunHO1gVs
Examplebucket,object%20key%20with%20spaces,9HkPvDaZY5MVbMhn6TMn1YTb5ArQAo3w
```

Menentukan file manifes yang ada

Anda dapat menentukan manifes untuk permintaan buat pekerjaan dengan menggunakan salah satu dari dua format berikut:

- Laporan Inventaris Amazon S3–Harus berupa laporan Inventaris Amazon S3 berformat CSV. Anda harus menentukan file `manifest.json` yang terkait dengan laporan inventaris. Untuk informasi selengkapnya tentang laporan inventaris, lihat Inventaris Amazon S3. Jika laporan inventaris menyertakan ID versi, Operasi Batch S3 beroperasi pada versi objek tertentu.

> **Note**
>
> - Operasi Batch S3 mendukung laporan inventaris CSV yang dienkripsi dengan SSE-KMS.
> - Jika Anda mengirimkan manifes laporan inventaris yang dienkripsi dengan SSE-KMS, kebijakan IAM Anda harus menyertakan izin `"kms:Decrypt"` dan `"kms:GenerateDataKey"` untuk objek `manifest.json` serta semua file data CSV terkait.

- File CSV–Setiap baris dalam file harus menyertakan nama bucket, kunci objek, dan secara opsional, versi objek. Kunci objek harus dienkode URL, seperti yang ditunjukkan dalam contoh berikut. Manifes harus menyertakan ID versi untuk semua objek atau menghilangkan ID versi untuk semua objek. Untuk informasi selengkapnya tentang format manifes CSV, lihat JobManifestSpec di Referensi API Amazon Simple Storage Service.

> **Note**
>
> Operasi Batch S3 tidak mendukung file manifes CSV yang dienkripsi dengan SSE-KMS.

> **Important**
>
> Saat Anda menggunakan manifes yang dibuat secara manual dan bucket yang memiliki versi, sebaiknya Anda menentukan ID versi untuk objek tersebut. Saat Anda membuat pekerjaan, Operasi Batch S3 mengurai seluruh manifes sebelum menjalankan pekerjaan tersebut. Namun, proses tersebut tidak mengambil "snapshot" dari status bucket.
> Karena manifes dapat berisi miliaran objek, pekerjaan mungkin membutuhkan waktu yang lama untuk dijalankan, sehingga dapat memengaruhi versi objek yang ditindaklanjuti oleh pekerjaan tersebut. Misalnya, Anda menimpa objek dengan versi baru saat pekerjaan berjalan dan Anda tidak menentukan ID versi untuk objek tersebut. Dalam hal ini, Amazon S3 akan melakukan operasi pada versi terbaru dari objek tersebut, bukan pada versi yang ada ketika Anda membuat pekerjaan. Satu-satunya cara untuk menghindari hal ini adalah dengan menentukan ID versi untuk objek yang terdaftar dalam manifes.

Membuat manifes secara otomatis

Anda dapat mengatur Amazon S3 untuk membuat manifes secara otomatis berdasarkan kriteria filter objek yang Anda tentukan saat membuat pekerjaan. Opsi ini tersedia untuk pekerjaan replikasi batch yang Anda buat di konsol Amazon S3, atau untuk jenis pekerjaan apa pun yang Anda buat dengan menggunakan AWS CLI AWS , SDK, atau Amazon S3 REST API. Untuk informasi selengkapnya tentang Replikasi Batch, lihat Mereplikasi objek yang ada dengan Replikasi Batch S3.

Untuk membuat manifes secara otomatis, tentukan elemen berikut sebagai bagian dari permintaan pembuatan pekerjaan Anda:

- Informasi tentang bucket yang berisi objek sumber Anda, termasuk pemilik bucket dan Amazon Resource Name (ARN)
- Informasi tentang output manifes, termasuk tanda untuk membuat file manifes, pemilik bucket output, ARN, prefiks, format file, dan jenis enkripsi
- Kriteria opsional untuk memfilter objek berdasarkan tanggal pembuatan, nama kunci, ukuran, kelas penyimpanan, dan tanda

Kriteria filter objek

Untuk memfilter daftar objek yang akan disertakan dalam manifes yang dibuat secara otomatis, Anda dapat menentukan kriteria berikut. Untuk informasi selengkapnya, lihat JobManifestGeneratorFilter di Referensi API Amazon S3.

CreatedAfter

Jika tersedia, manifes yang dibuat hanya akan mencakup objek bucket sumber yang dibuat setelah waktu ini.

CreatedBefore

Jika tersedia, manifes yang dibuat hanya akan mencakup objek bucket sumber yang dibuat sebelum waktu ini.

EligibleForReplication

Jika tersedia, manifes yang dibuat akan mencakup objek hanya jika objek tersebut memenuhi syarat untuk direplikasi sesuai dengan konfigurasi replikasi pada bucket sumber.

KeyNameConstraint

Jika disediakan, manifes yang dihasilkan hanya mencakup objek bucket sumber yang kunci objeknya cocok dengan batasan string yang ditentukan untuk MatchAnySubstring,, MatchAnyPrefixdan. MatchAnySuffix

MatchAnySubstring— Jika disediakan, manifes yang dihasilkan mencakup objek jika string yang ditentukan muncul di mana saja dalam string kunci objek.

MatchAnyPrefix— Jika disediakan, manifes yang dihasilkan mencakup objek jika string yang ditentukan muncul di awal string kunci objek.

MatchAnySuffix— Jika disediakan, manifes yang dihasilkan mencakup objek jika string yang ditentukan muncul di akhir string kunci objek.

MatchAnyStorageClass

Jika tersedia, manifes yang dibuat hanya akan mencakup objek bucket sumber yang disimpan dengan kelas penyimpanan yang ditentukan.

ObjectReplicationStatuses

Jika tersedia, manifes yang dibuat hanya akan mencakup objek bucket sumber yang memiliki salah satu status replikasi yang ditentukan.

ObjectSizeGreaterThanBytes

Jika tersedia, manifes yang dibuat hanya akan mencakup objek bucket sumber yang ukuran filenya lebih besar dari jumlah byte yang ditentukan.

ObjectSizeLessThanBytes

Jika tersedia, manifes yang dibuat hanya akan mencakup objek bucket sumber yang ukuran filenya kurang dari jumlah byte yang ditentukan.

> **Note**
>
> Anda tidak dapat mengkloning sebagian besar tugas yang telah membuat manifes secara otomatis. Pekerjaan replikasi batch dapat dikloning, kecuali ketika pekerjaan tersebut menggunakan kriteria filter manifes `KeyNameConstraint`, `MatchAnyStorageClass`, `ObjectSizeGreaterThanBytes`, atau `ObjectSizeLessThanBytes`.

Sintaks untuk menentukan kriteria manifes bervariasi tergantung pada metode yang Anda gunakan untuk membuat pekerjaan. Sebagai contoh, lihat Membuat pekerjaan.

Membuat pekerjaan

Anda dapat membuat pekerjaan Operasi Batch S3 dengan menggunakan konsol Amazon S3, SDK AWS CLI AWS , atau Amazon S3 REST API.

Untuk informasi selengkapnya tentang cara membuat permintaan pekerjaan, lihat Elemen permintaan pekerjaan Operasi Batch.

Prasyarat

Sebelum membuat pekerjaan Operasi Batch, konfirmasikan bahwa Anda telah mengonfigurasi izin yang relevan. Untuk informasi selengkapnya, lihat Memberikan izin untuk Operasi Batch Amazon S3.

Menggunakan konsol S3

Untuk membuat pekerjaan batch

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di bilah navigasi di bagian atas halaman, pilih nama yang saat ini ditampilkan Wilayah AWS. Selanjutnya, pilih Wilayah tempat Anda ingin membuat pekerjaan Anda.

   > **Note**
   >
   > Untuk operasi penyalinan, Anda harus membuat pekerjaan di Wilayah yang sama dengan bucket tujuan. Untuk semua operasi lainnya, Anda harus membuat pekerjaan di Wilayah yang sama dengan objek dalam manifes.

3. Pilih Operasi Batch di panel navigasi kiri konsol Amazon S3.
4. Pilih Buat tugas.
5. Lihat di Wilayah AWSmana Anda ingin membuat pekerjaan Anda.
6. Pada Format manifes, pilih jenis objek manifes yang akan digunakan.

   - Jika Anda memilih Laporan inventaris S3, masukkan jalur ke objek manifes.json yang dihasilkan Amazon S3 sebagai bagian dari laporan Inventaris berformat CSV, dan secara opsional, ID versi untuk objek manifes jika Anda ingin menggunakan versi selain yang terbaru.

- Jika Anda memilih CSV, masukkan jalur ke objek manifes yang diformat CSV. Objek manifes harus mengikuti format yang dijelaskan di konsol. Anda dapat memasukkan ID versi secara opsional untuk objek manifes jika ingin menggunakan versi selain yang terbaru.

> **Note**
>
> Konsol Amazon S3 mendukung pembuatan manifes otomatis untuk pekerjaan replikasi batch saja. Untuk semua jenis pekerjaan lainnya, jika Anda ingin Amazon S3 menghasilkan manifes secara otomatis berdasarkan kriteria filter yang Anda tentukan, Anda harus mengonfigurasi pekerjaan menggunakan, AWS SDK AWS CLI, atau Amazon S3 REST API.

7. Pilih Selanjutnya.
8. Pada Operasi, pilih operasi yang ingin Anda jalankan pada semua objek yang tercantum dalam manifes. Isi informasi untuk operasi yang Anda pilih lalu pilih Selanjutnya.
9. Isi informasi untuk Konfigurasikan opsi tambahan, lalu pilih Selanjutnya.
10. Untuk Peninjauan, verifikasikan pengaturan. Jika Anda perlu membuat perubahan, pilih Sebelumnya. Atau, pilih Buat Tugas.

Menggunakan AWS CLI

Specify manifest

Contoh berikut menunjukkan cara membuat pekerjaan `S3PutObjectTagging` Operasi Batch S3 yang bekerja pada objek yang terdaftar dalam file manifes yang ada.

Untuk membuat pekerjaan **`S3PutObjectTagging`** Operasi Batch

1. Gunakan perintah berikut untuk membuat peran AWS Identity and Access Management (IAM), lalu buat kebijakan IAM untuk menetapkan izin yang relevan. Peran dan kebijakan berikut memberikan izin Amazon S3 untuk menambahkan tanda objek yang akan Anda perlukan saat membuat pekerjaan di langkah berikutnya.

   a. Gunakan perintah contoh berikut untuk membuat peran IAM untuk Operasi Batch yang akan digunakan. Untuk menggunakan contoh perintah ini, ganti *`S3BatchJobRole`* dengan nama yang ingin Anda berikan ke peran tersebut.

```
aws iam create-role \
 --role-name S3BatchJobRole \
 --assume-role-policy-document '{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"batchoperations.s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}'
```

Catat Amazon Resource Name (ARN) untuk peran. Anda akan membutuhkan ARN saat membuat pekerjaan.

b. Gunakan contoh berikut untuk membuat kebijakan IAM dengan izin yang diperlukan dan lampirkan ke peran IAM yang Anda buat di langkah sebelumnya. Untuk informasi selengkapnya tentang izin yang diperlukan, lihat Memberikan izin untuk Operasi Batch Amazon S3.

> **Note**
>
> Pekerjaan Operasi Batch yang menjalankan tindakan pada bucket direktori memerlukan izin tertentu. Untuk informasi selengkapnya, lihat AWS Identity and Access Management (IAM) untuk S3 Express One Zone.

Untuk menggunakan perintah contoh ini, ganti *user input placeholders* sebagai berikut:

- Ganti *S3BatchJobRole* dengan nama peran IAM Anda. Pastikan nama ini cocok dengan nama yang Anda gunakan sebelumnya.
- Ganti *PutObjectTaggingBatchJobPolicy* dengan nama yang ingin Anda berikan pada kebijakan IAM.
- Ganti *example-s3-destination-bucket* dengan nama bucket yang berisi objek yang ingin Anda berikan tanda.

- Ganti *DOC-EXAMPLE-MANIFEST-BUCKET* dengan nama bucket yang berisi manifes.
- Ganti *DOC-EXAMPLE-REPORT-BUCKET* dengan nama bucket tujuan pengiriman laporan penyelesaian.

```
aws iam put-role-policy \
  --role-name S3BatchJobRole \
  --policy-name PutObjectTaggingBatchJobPolicy \
  --policy-document '{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObjectTagging",
        "s3:PutObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::example-s3-destination-bucket/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-BUCKET/*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObject",
        "s3:GetBucketLocation"
      ],
      "Resource":[
        "arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET/*"
      ]
    }
```

```
  ]
}'
```

2. Gunakan contoh perintah berikut untuk membuat pekerjaan `S3PutObjectTagging`.

   File `manifest.csv` menyediakan daftar bucket dan nilai kunci objek. Pekerjaan ini menerapkan tanda yang ditentukan ke objek yang diidentifikasi dalam manifes. `ETag` adalah ETag objek `manifest.csv` yang bisa Anda dapatkan dari konsol Amazon S3. Permintaan ini menentukan parameter `no-confirmation-required`, sehingga Anda dapat menjalankan pekerjaan tanpa harus mengonfirmasinya dengan perintah `update-job-status`. Untuk informasi selengkapnya, lihat create-job di AWS CLI Referensi Perintah.

   Untuk menggunakan perintah contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri. Ganti *`IAM-role`* dengan ARN peran IAM yang Anda buat sebelumnya.

```
aws s3control create-job \
    --region us-west-2 \
    --account-id acct-id \
    --operation '{"S3PutObjectTagging": { "TagSet": [{"Key":"keyOne",
 "Value":"ValueOne"}] }}' \
    --manifest '{"Spec":{"Format":"S3BatchOperations_CSV_20180820","Fields":
["Bucket","Key"]},"Location":
{"ObjectArn":"arn:aws:s3:::my_manifests/
manifest.csv","ETag":"60e460c9d1046e73f7dde5043ac3ae85"}}' \
    --report '{"Bucket":"arn:aws:s3:::DOC-EXAMPLE-REPORT-
BUCKET","Prefix":"final-reports",
 "Format":"Report_CSV_20180820","Enabled":true,"ReportScope":"AllTasks"}' \
    --priority 42 \
    --role-arn IAM-role \
    --client-request-token $(uuidgen) \
    --description "job description" \
    --no-confirmation-required
```

   Sebagai tanggapan, Amazon S3 akan mengembalikan ID pekerjaan (misalnya, `00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c`). Anda akan memerlukan ID pekerjaan untuk mengidentifikasi, memantau, dan mengubah pekerjaan.

Generate manifest

Contoh berikut menunjukkan cara membuat pekerjaan `S3DeleteObjectTagging` Operasi Batch S3 yang secara otomatis membuat manifes berdasarkan kriteria filter objek Anda. Kriteria ini mencakup tanggal pembuatan, nama kunci, ukuran, kelas penyimpanan, dan tanda.

Untuk membuat pekerjaan **`S3DeleteObjectTagging`** Operasi Batch

1. Gunakan perintah berikut untuk membuat peran AWS Identity and Access Management (IAM), lalu buat kebijakan IAM untuk menetapkan izin. Peran dan kebijakan berikut memberikan izin Amazon S3 untuk menghapus tanda objek, yang akan Anda perlukan saat membuat pekerjaan di langkah berikutnya.

   a. Gunakan perintah contoh berikut untuk membuat peran IAM untuk Operasi Batch yang akan digunakan. Untuk menggunakan contoh perintah ini, ganti *`S3BatchJobRole`* dengan nama yang ingin Anda berikan ke peran tersebut.

```
aws iam create-role \
 --role-name S3BatchJobRole \
 --assume-role-policy-document '{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"batchoperations.s3.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}'
```

   Catat Amazon Resource Name (ARN) untuk peran. Anda akan membutuhkan ARN saat membuat pekerjaan.

   b. Gunakan contoh berikut untuk membuat kebijakan IAM dengan izin yang diperlukan dan lampirkan ke peran IAM yang Anda buat di langkah sebelumnya. Untuk informasi selengkapnya tentang izin yang diperlukan, lihat Memberikan izin untuk Operasi Batch Amazon S3.

> **Note**
>
> Pekerjaan Operasi Batch yang menjalankan tindakan pada bucket direktori memerlukan izin tertentu. Untuk informasi selengkapnya, lihat AWS Identity and Access Management (IAM) untuk S3 Express One Zone.

Untuk menggunakan perintah contoh ini, ganti *user input placeholders* sebagai berikut:

- Ganti *S3BatchJobRole* dengan nama peran IAM Anda. Pastikan nama ini cocok dengan nama yang Anda gunakan sebelumnya.
- Ganti *DeleteObjectTaggingBatchJobPolicy* dengan nama yang ingin Anda berikan pada kebijakan IAM.
- Ganti *example-s3-destination-bucket* dengan nama bucket yang berisi objek yang ingin Anda berikan tanda.
- Ganti *DOC-EXAMPLE-MANIFEST-OUTPUT-BUCKET* dengan nama bucket tempat Anda ingin menyimpan manifes.
- Ganti *DOC-EXAMPLE-REPORT-BUCKET* dengan nama bucket tujuan pengiriman laporan penyelesaian.

```
aws iam put-role-policy \
  --role-name S3BatchJobRole \
  --policy-name DeleteObjectTaggingBatchJobPolicy \
  --policy-document '{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:DeleteObjectTagging",
        "s3:DeleteObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::example-s3-destination-bucket/*"
    },
    {
      "Effect":"Allow",
```

```
      "Action":[
        "s3:PutInventoryConfiguration"
      ],
      "Resource": "arn:aws:s3:::example-s3-destination-bucket"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-OUTPUT-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-OUTPUT-BUCKET/*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:PutObject",
        "s3:ListBucket"
      ],
      "Resource":[
        "arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET/*",
        "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-OUTPUT-BUCKET/*"
      ]
    }
  ]
}'
```

2. Gunakan contoh berikut untuk membuat pekerjaan `S3DeleteObjectTagging`.

   Dalam contoh ini, nilai di bagian `--report` menentukan bucket, prefiks, format, dan cakupan laporan pekerjaan yang akan dihasilkan. Bagian `--manifest -generator` ini menentukan informasi tentang bucket sumber yang berisi objek yang akan ditindaklanjuti oleh pekerjaan, informasi tentang daftar output manifes yang akan dibuat untuk pekerjaan, dan kriteria filter untuk mempersempit cakupan objek yang akan disertakan dalam manifes berdasarkan tanggal pembuatan, batasan nama, ukuran, dan kelas penyimpanan. Perintah ini juga menentukan prioritas pekerjaan, peran IAM, dan Wilayah AWS.

Untuk informasi selengkapnya, lihat create-job di AWS CLI Referensi Perintah.

Untuk menggunakan perintah contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri. Ganti *IAM-role* dengan ARN peran IAM yang Anda buat sebelumnya.

```
aws s3control create-job \
    --account-id 012345678901 \
    --operation '{
        "S3DeleteObjectTagging": {}
    }' \
    --report '{
        "Bucket":"arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET",
        "Prefix":"reports",
        "Format":"Report_CSV_20180820",
        "Enabled":true,
        "ReportScope":"AllTasks"
    }' \
    --manifest-generator '{
        "S3JobManifestGenerator": {
          "ExpectedBucketOwner": "012345678901",
          "SourceBucket": "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET",
          "EnableManifestOutput": true,
          "ManifestOutputLocation": {
            "ExpectedManifestBucketOwner": "012345678901",
            "Bucket": "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-OUTPUT-BUCKET",
            "ManifestPrefix": "prefix",
            "ManifestFormat": "S3InventoryReport_CSV_20211130"
          },
          "Filter": {
            "CreatedAfter": "2023-09-01",
            "CreatedBefore": "2023-10-01",
            "KeyNameConstraint": {
              "MatchAnyPrefix": [
                "prefix"
              ],
              "MatchAnySuffix": [
                "suffix"
              ]
            },
            "ObjectSizeGreaterThanBytes": 100,
            "ObjectSizeLessThanBytes": 200,
```

```
            "MatchAnyStorageClass": [
              "STANDARD",
              "STANDARD_IA"
            ]
          }
        }
      }' \
      --priority 2 \
      --role-arn IAM-role \
      --region us-east-1
```

Sebagai tanggapan, Amazon S3 akan mengembalikan ID pekerjaan (misalnya, `00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c`). Anda akan memerlukan ID pekerjaan ini untuk mengidentifikasi, memantau, atau mengubah pekerjaan.

Menggunakan AWS SDK for Java

Specify manifest

Contoh berikut menunjukkan cara membuat pekerjaan `S3PutObjectTagging` Operasi Batch S3 yang bekerja pada objek yang terdaftar dalam file manifes yang ada. Untuk menggunakan contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

Example

```
package aws.example.s3control;



import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.*;

import java.util.UUID;
import java.util.ArrayList;

import static com.amazonaws.regions.Regions.US_WEST_2;
```

```
public class CreateJob {
    public static void main(String[] args) {
        String accountId = "Account ID";
        String iamRoleArn = "IAM Role ARN";
        String reportBucketName = "arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET";
        String uuid = UUID.randomUUID().toString();

        ArrayList tagSet = new ArrayList<S3Tag>();
        tagSet.add(new S3Tag().withKey("keyOne").withValue("ValueOne"));


        try {
            JobOperation jobOperation = new JobOperation()
                    .withS3PutObjectTagging(new S3SetObjectTaggingOperation()
                            .withTagSet(tagSet)
                    );

            JobManifest manifest = new JobManifest()
                    .withSpec(new JobManifestSpec()
                            .withFormat("S3BatchOperations_CSV_20180820")
                            .withFields(new String[]{
                                    "Bucket", "Key"
                            }))
                    .withLocation(new JobManifestLocation()
                            .withObjectArn("arn:aws:s3:::my_manifests/manifest.csv")
                            .withETag("60e460c9d1046e73f7dde5043ac3ae85"));
            JobReport jobReport = new JobReport()
                    .withBucket(reportBucketName)
                    .withPrefix("reports")
                    .withFormat("Report_CSV_20180820")
                    .withEnabled(true)
                    .withReportScope("AllTasks");

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.createJob(new CreateJobRequest()
                    .withAccountId(accountId)
                    .withOperation(jobOperation)
                    .withManifest(manifest)
                    .withReport(jobReport)
                    .withPriority(42)
```

```
                    .withRoleArn(iamRoleArn)
                    .withClientRequestToken(uuid)
                    .withDescription("job description")
                    .withConfirmationRequired(false)
            );

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Generate manifest

Contoh berikut menunjukkan cara membuat pekerjaan `s3PutObjectCopy` Operasi Batch S3 yang secara otomatis membuat manifes berdasarkan kriteria filter objek, termasuk tanggal pembuatan, nama kunci, dan ukuran. Untuk menggunakan contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

Example

```
        package aws.example.s3control;

    import com.amazonaws.AmazonServiceException;
    import com.amazonaws.SdkClientException;
    import com.amazonaws.auth.profile.ProfileCredentialsProvider;
    import com.amazonaws.services.s3control.AWSS3Control;
    import com.amazonaws.services.s3control.AWSS3ControlClient;
    import com.amazonaws.services.s3control.model.CreateJobRequest;
    import com.amazonaws.services.s3control.model.CreateJobResult;
    import com.amazonaws.services.s3control.model.JobManifestGenerator;
    import com.amazonaws.services.s3control.model.JobManifestGeneratorFilter;
    import com.amazonaws.services.s3control.model.JobOperation;
    import com.amazonaws.services.s3control.model.JobReport;
    import com.amazonaws.services.s3control.model.KeyNameConstraint;
    import com.amazonaws.services.s3control.model.S3JobManifestGenerator;
```

```
    import com.amazonaws.services.s3control.model.S3ManifestOutputLocation;
    import com.amazonaws.services.s3control.model.S3SetObjectTaggingOperation;
    import com.amazonaws.services.s3control.model.S3Tag;

    import java.time.Instant;
    import java.util.Date;
    import java.util.UUID;
    import java.util.ArrayList;

    import static com.amazonaws.regions.Regions.US_WEST_2;

    public class test {
        public static void main(String[] args) {
            String accountId = "012345678901";
            String iamRoleArn = "arn:aws:iam::012345678901:role/ROLE";
            String sourceBucketName = "arn:aws:s3:::DOC-EXAMPLE-SOURCE-BUCKET";
            String reportBucketName = "arn:aws:s3:::DOC-EXAMPLE-REPORT-BUCKET";
            String manifestOutputBucketName = "arn:aws:s3:::DOC-EXAMPLE-MANIFEST-
OUTPUT-BUCKET";
            String uuid = UUID.randomUUID().toString();
            long minimumObjectSize = 100L;

            ArrayList<S3Tag> tagSet = new ArrayList<>();
            tagSet.add(new S3Tag().withKey("keyOne").withValue("ValueOne"));

            ArrayList<String> prefixes = new ArrayList<>();
            prefixes.add("s3KeyStartsWith");

            try {
                JobOperation jobOperation = new JobOperation()
                        .withS3PutObjectTagging(new S3SetObjectTaggingOperation()
                                .withTagSet(tagSet)
                        );
                S3ManifestOutputLocation manifestOutputLocation = new
 S3ManifestOutputLocation()
                        .withBucket(manifestOutputBucketName)
                        .withManifestPrefix("manifests")
                        .withExpectedManifestBucketOwner(accountId)
                        .withManifestFormat("S3InventoryReport_CSV_20211130");

                JobManifestGeneratorFilter jobManifestGeneratorFilter = new
 JobManifestGeneratorFilter()
                        .withEligibleForReplication(true)
                        .withKeyNameConstraint(
```

```
                                new KeyNameConstraint()
                                        .withMatchAnyPrefix(prefixes))
                        .withCreatedBefore(Date.from(Instant.now()))
                        .withObjectSizeGreaterThanBytes(minimumObjectSize);

                S3JobManifestGenerator s3JobManifestGenerator = new
S3JobManifestGenerator()
                        .withEnableManifestOutput(true)
                        .withManifestOutputLocation(manifestOutputLocation)
                        .withFilter(jobManifestGeneratorFilter)
                        .withSourceBucket(sourceBucketName);

                JobManifestGenerator jobManifestGenerator = new
JobManifestGenerator()
                        .withS3JobManifestGenerator(s3JobManifestGenerator);

                JobReport jobReport = new JobReport()
                        .withBucket(reportBucketName)
                        .withPrefix("reports")
                        .withFormat("Report_CSV_20180820")
                        .withEnabled(true)
                        .withReportScope("AllTasks");

                AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                        .withCredentials(new ProfileCredentialsProvider())
                        .withRegion(US_WEST_2)
                        .build();

                CreateJobResult createJobResult = s3ControlClient.createJob(new
CreateJobRequest()
                        .withAccountId(accountId)
                        .withOperation(jobOperation)
                        .withManifestGenerator(jobManifestGenerator)
                        .withReport(jobReport)
                        .withPriority(42)
                        .withRoleArn(iamRoleArn)
                        .withClientRequestToken(uuid)
                        .withDescription("job description")
                        .withConfirmationRequired(true)
                );

                System.out.println("Created job " + createJobResult.getJobId());

            } catch (AmazonServiceException e) {
```

```
                // The call was transmitted successfully, but Amazon S3 couldn't
process
                // it and returned an error response.
                e.printStackTrace();
            } catch (SdkClientException e) {
                // Amazon S3 couldn't be contacted for a response, or the client
                // couldn't parse the response from Amazon S3.
                e.printStackTrace();
            }
        }
    }
```

Penggunaan API REST

Anda dapat menggunakan API REST untuk membuat pekerjaan Operasi Batch. Untuk informasi selengkapnya, lihat CreateJob di Referensi API Amazon Simple Storage Service.

Respons pekerjaan

Jika permintaan `CreateJob` berhasil, Amazon S3 akan mengembalikan ID pekerjaan. ID pekerjaan adalah pengenal unik yang dihasilkan secara otomatis oleh Amazon S3 sehingga Anda dapat mengidentifikasi pekerjaan Operasi Batch dan memantau statusnya.

Saat Anda membuat pekerjaan melalui AWS CLI, AWS SDK, atau REST API, Anda dapat menyetel Operasi Batch S3 untuk mulai memproses pekerjaan secara otomatis. Pekerjaan akan segera dijalankan setelah siap, tanpa harus menunggu pekerjaan yang lebih diprioritaskan.

Saat membuat pekerjaan melalui konsol Amazon S3, Anda harus meninjau detail pekerjaan dan mengonfirmasi bahwa Anda ingin menjalankannya sebelum Operasi Batch dapat mulai memprosesnya. Jika suatu pekerjaan tetap berada dalam status ditangguhkan selama lebih dari 30 hari, pekerjaan tersebut akan gagal.

Operasi yang didukung oleh Operasi Batch S3

S3 Batch Operations mendukung beberapa operasi yang berbeda. Topik dalam bagian ini menjelaskan masing-masing operasi tersebut.

Salin objek

Operasi Salin menyalin setiap objek yang ditentukan dalam manifes. Anda dapat menyalin objek ke bucket di Wilayah AWS yang sama atau ke bucket di Wilayah berbeda. S3 Batch Operations

mendukung sebagian besar opsi yang tersedia melalui Amazon S3 untuk menyalin objek. Opsi ini termasuk mengatur metadata objek, mengatur izin, dan mengubah kelas penyimpanan objek.

Anda juga dapat menggunakan operasi Salin untuk menyalin objek tidak terenkripsi yang ada dan menulisnya kembali ke bucket yang sama dengan objek terenkripsi. Untuk informasi selengkapnya, lihat Mengenkripsi objek dengan Operasi Batch Amazon S3.

Ketika Anda menyalin objek, Anda dapat mengubah algoritma checksum yang digunakan untuk menghitung checksum objek. Jika objek tidak memiliki checksum tambahan yang dihitung, Anda juga dapat menambahkannya dengan menentukan algoritma checksum untuk Amazon S3 untuk digunakan. Untuk informasi selengkapnya, lihat Memeriksa integritas objek.

Untuk informasi lebih lanjut tentang menyalin objek di Amazon S3 dan parameter opsional yang diperlukan, lihat Menyalin, memindahkan, dan mengganti nama objek dalam panduan ini dan CopyObject diReferensi API Amazon Simple Storage Service.

Pembatasan dan batasan

- Semua objek sumber harus berada dalam satu bucket.
- Semua objek tujuan harus berada dalam satu bucket.
- Anda harus memiliki izin baca untuk bucket sumber dan izin tulis untuk bucket tujuan.
- Objek yang akan disalin dapat berukuran hingga 5 GB.
- Jika Anda mencoba menyalin objek dari kelas S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive ke kelas penyimpanan Standar S3, Anda harus terlebih dahulu memulihkan objek ini. Untuk informasi selengkapnya, lihat Memulihkan objek yang diarsipkan.
- Tugas penyalinan harus dibuat di Wilayah tujuan, yang merupakan Wilayah tujuan untuk menyalin objek.
- Semua opsi Salin didukung kecuali untuk pemeriksaan bersyarat pada ETag dan enkripsi sisi server dengan kunci enkripsi yang disediakan pelanggan (SSE-C).
- Jika bucket tidak berversi, Anda akan menimpa objek dengan nama kunci yang sama.
- Objek tidak selalu disalin dalam urutan yang sama seperti yang tercantum dalam manifes. Untuk bucket berversi, jika mempertahankan urutan versi saat ini/tidak-saat ini penting, Anda harus menyalin semua versi tidak terbaru terlebih dahulu. Kemudian, setelah pekerjaan pertama selesai, salin versi terbaru pada pekerjaan berikutnya.
- Menyalin objek ke kelas Reduced Redundancy Storage (RRS) tidak didukung.

Menyalin objek menggunakan Operasi Batch S3

Anda dapat menggunakan Operasi Batch S3 untuk membuat pekerjaan salin PUT untuk menyalin objek dalam akun yang sama atau ke akun tujuan yang berbeda. Bagian berikut berisi contoh cara menyimpan dan menggunakan manifes yang ada di akun yang berbeda. Di bagian pertama, Anda dapat menggunakan Amazon S3 Inventory untuk mengirimkan laporan inventaris ke akun tujuan untuk digunakan selama pembuatan tugas atau, Anda dapat menggunakan manifes nilai yang dipisahkan koma (CSV) di akun sumber atau tujuan, seperti yang ditunjukkan dalam contoh kedua. Contoh ketiga menunjukkan bagaimana menggunakan operasi Salin untuk mengaktifkan enkripsi Kunci Bucket S3 pada objek yang sudah ada.

Contoh Operasi Salin

- Menggunakan laporan inventaris yang dikirimkan ke akun tujuan untuk menyalin objek di seluruh Akun AWS
- Menggunakan manifes CSV yang disimpan di akun sumber untuk menyalin objek di seluruh Akun AWS
- Menggunakan Operasi Batch S3 untuk mengenkripsi objek dengan Kunci Bucket S3

Menggunakan laporan inventaris yang dikirimkan ke akun tujuan untuk menyalin objek di seluruh Akun AWS

Anda dapat menggunakan Amazon S3 Inventory untuk membuat laporan inventaris dan menggunakan laporan untuk membuat daftar objek untuk menyalin dengan Operasi Batch S3. Untuk informasi selengkapnya tentang menggunakan manifes CSV di akun sumber atau tujuan, lihat the section called “Menggunakan manifes CSV untuk menyalin objek di seluruh Akun AWS”.

Amazon S3 Inventory menghasilkan inventaris objek dalam bucket. Daftar yang dihasilkan diterbitkan ke file output. Kategori yang diinventarisasi disebut bucket sumber, dan bucket tempat menyimpan file laporan inventaris disebut sebagai bucket tujuan.

Laporan inventaris Amazon S3 dapat dikonfigurasi untuk dikirimkan ke yang lainAkun AWS. Ini memungkinkan Operasi Batch S3 untuk membaca laporan inventaris saat pekerjaan dibuat di akun tujuan.

Untuk informasi lebih lanjut tentang bucket sumber dan tujuan Amazon S3 Inventory, lihat Bucket sumber dan tujuan.

Cara termudah untuk menyiapkan inventaris adalah dengan menggunakan AWS Management Console, tetapi Anda juga dapat menggunakan REST API, AWS Command Line Interface (AWS CLI), atau AWS SDK.

Prosedur konsol berikut berisi langkah-langkah tingkat tinggi untuk mengatur izin bagi pekerjaan S3 Batch Operations. Dalam prosedur ini, Anda menyalin objek dari akun sumber ke akun tujuan, dengan laporan inventaris yang disimpan di akun tujuan.

Untuk menyiapkan inventaris Amazon S3 untuk bucket sumber dan tujuan yang dimiliki oleh akun yang berbeda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pilih bucket tujuan untuk menyimpan laporan inventaris.

   Tentukan bucket manifest tujuan untuk menyimpan laporan inventaris. Dalam prosedur ini, akun tujuan adalah akun yang memiliki bucket manifest tujuan dan bucket yang menerima salinan objek.
3. Konfigurasikan inventaris untuk mencantumkan objek dalam bucket sumber dan publikasikan daftar ke bucket manifest tujuan.

   Konfigurasikan daftar inventaris untuk bucket sumber. Saat melakukan ini, Anda menentukan bucket tujuan tempat Anda ingin menyimpan daftar. Laporan inventaris untuk bucket sumber diterbitkan ke bucket tujuan. Dalam prosedur ini, akun sumber adalah akun yang memiliki bucket sumber.

   Untuk informasi tentang cara menggunakan konsol untuk mengonfigurasi inventaris, lihatMengonfigurasi Inventaris Amazon S3.

   Pilih CSV untuk format output.

   Saat Anda memasukkan informasi untuk bucket tujuan, pilih Bucket di akun lain. Kemudian masukkan nama bucket manifest tujuan. Atau, Anda dapat memasukkan ID akun dari akun tujuan.

   Setelah konfigurasi inventaris disimpan, konsol menampilkan pesan yang serupa dengan pesan berikut:

Amazon S3 tidak dapat membuat kebijakan bucket pada bucket tujuan. Meminta pemilik bucket tujuan untuk menambahkan kebijakan bucket berikut agar Amazon S3 menempatkan data dalam bucket tersebut.

Konsol kemudian menampilkan kebijakan bucket yang dapat Anda gunakan untuk bucket tujuan.

4. Salin kebijakan bucket tujuan yang muncul di konsol.
5. Pada akun tujuan, tambahkan kebijakan bucket yang disalin ke bucket manifest tujuan tempat penyimpanan laporan inventaris.
6. Buat peran di akun tujuan yang didasarkan pada kebijakan kepercayaan S3 Batch Operations. Untuk informasi lebih lanjut tentang kebijakan kepercayaan, lihat Kebijakan kepercayaan.

   Untuk informasi selengkapnya tentang membuat peran, lihat Membuat peran untuk mendelegasikan izin ke layanan AWS di Panduan Pengguna IAM.

   Masukkan nama untuk peran (peran contoh menggunakan nama `BatchOperationsDestinationRoleCOPY`). Pilih layanan S3, lalu pilih kasus penggunaan bucket S3 Batch Operations, yang menerapkan kebijakan kepercayaan pada peran tersebut.

   Kemudian pilih Buat kebijakan untuk melampirkan kebijakan berikut pada peran tersebut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowBatchOperationsDestinationObjectCOPY",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:PutObjectVersionAcl",
        "s3:PutObjectAcl",
        "s3:PutObjectVersionTagging",
        "s3:PutObjectTagging",
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetObjectAcl",
        "s3:GetObjectTagging",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionTagging"
      ],
```

```
      "Resource": [
        "arn:aws:s3:::ObjectDestinationBucket/*",
        "arn:aws:s3:::ObjectSourceBucket/*",
        "arn:aws:s3:::ObjectDestinationManifestBucket/*"
      ]
    }
  ]
}
```

Peran menggunakan kebijakan untuk memberikan `batchoperations.s3.amazonaws.com` izin untuk membaca manifest di bucket tujuan. Peran ini juga memberikan izin untuk MENDAPATKAN (GET) objek, daftar kontrol akses (access control list/ACL), tag, dan versi di bucket objek sumber. Dan itu memberikan izin untuk PUT objek, ACL, tanda, dan versi ke dalam bucket objek tujuan.

7. Di akun sumber, buat kebijakan bucket untuk bucket sumber yang memberikan peran yang Anda buat di langkah sebelumnya untuk GET objek, ACL, tanda, dan versi di bucket sumber. Langkah ini memungkinkan S3 Batch Operations mendapatkan objek dari bucket sumber melalui peran tepercaya.

   Berikut ini adalah contoh kebijakan bucket untuk akun sumber.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowBatchOperationsSourceObjectCOPY",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::DestinationAccountNumber:role/
BatchOperationsDestinationRoleCOPY"
      },
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetObjectAcl",
        "s3:GetObjectTagging",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::ObjectSourceBucket/*"
    }
```

```
  ]
}
```

8. Setelah laporan inventaris tersedia, buat pekerjaan penyalinan objek S3 Batch Operations PUT di akun tujuan, pilih laporan inventaris dari bucket manifest tujuan. Anda memerlukan ARN untuk peran yang Anda buat pada akun tujuan.

   Untuk informasi umum tentang cara membuat pekerjaan, lihat Membuat pekerjaan Operasi Batch S3.

   Untuk informasi tentang membuat tugas menggunakan konsol, lihat Membuat pekerjaan Operasi Batch S3.

Menggunakan manifes CSV yang disimpan di akun sumber untuk menyalin objek di seluruh Akun AWS

Anda dapat menggunakan file CSV yang disimpan di Akun AWS yang berbeda sebagai manifes untuk pekerjaan Operasi Batch S3. Untuk menggunakan S3 Inventory Report, lihat the section called "Menggunakan laporan inventaris untuk menyalin objek di seluruh Akun AWS".

Prosedur berikut menunjukkan cara menyiapkan izin saat menggunakan pekerjaan S3 Batch Operations untuk menyalin objek dari akun sumber ke akun tujuan dengan file manifest CSV yang tersimpan di akun sumber.

Untuk mengatur CSV manifest yang tersimpan di Akun AWS yang berbeda

1. Buat peran di akun tujuan yang didasarkan pada kebijakan kepercayaan S3 Batch Operations. Dalam prosedur ini, akun tujuan adalah akun tempat objek akan disalin.

   Untuk informasi lebih lanjut tentang kebijakan kepercayaan, lihat Kebijakan kepercayaan.

   Untuk informasi selengkapnya tentang membuat peran, lihat Membuat peran untuk mendelegasikan izin ke layanan AWS di Panduan Pengguna IAM.

   Jika Anda membuat peran menggunakan konsol, masukkan nama untuk peran tersebut (peran contoh menggunakan nama `BatchOperationsDestinationRoleCOPY`). Pilih layanan S3, lalu pilih kasus penggunaan bucket S3 Batch Operations, yang menerapkan kebijakan kepercayaan pada peran tersebut.

   Kemudian pilih Buat kebijakan untuk melampirkan kebijakan berikut pada peran tersebut.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowBatchOperationsDestinationObjectCOPY",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:PutObjectVersionAcl",
        "s3:PutObjectAcl",
        "s3:PutObjectVersionTagging",
        "s3:PutObjectTagging",
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetObjectAcl",
        "s3:GetObjectTagging",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionTagging"
      ],
      "Resource": [
        "arn:aws:s3:::ObjectDestinationBucket/*",
        "arn:aws:s3:::ObjectSourceBucket/*",
        "arn:aws:s3:::ObjectSourceManifestBucket/*"
      ]
    }
  ]
}
```

Menggunakan kebijakan, peran memberikan `batchoperations.s3.amazonaws.com` izin untuk membaca manifest di bucket manifest sumber. Ini memberikan izin untuk GET objek, ACL, tanda, dan versi di bucket objek sumber. Itu juga memberikan izin untuk PUT objek, ACL, tanda, dan versi ke dalam bucket objek tujuan.

2. Pada akun sumber, buat kebijakan bucket untuk bucket yang berisi manifest untuk memberikan peran yang Anda buat pada langkah sebelumnya untuk MENDAPATKAN (GET) objek dan versi di bucket manifest sumber.

   Langkah ini memungkinkan S3 Batch Operations untuk membaca manifest menggunakan peran tepercaya. Terapkan kebijakan bucket ke bucket yang berisi manifest.

   Berikut ini adalah contoh kebijakan bucket untuk diterapkan pada bucket manifest sumber.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowBatchOperationsSourceManfiestRead",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::DestinationAccountNumber:user/ConsoleUserCreatingJob",
          "arn:aws:iam::DestinationAccountNumber:role/
BatchOperationsDestinationRoleCOPY"
        ]
      },
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": "arn:aws:s3:::ObjectSourceManifestBucket/*"
    }
  ]
}
```

Kebijakan ini juga memberikan izin untuk memungkinkan pengguna konsol yang membuat pekerjaan pada akun tujuan, menggunakan izin yang sama dalam bucket manifest sumber, melalui kebijakan bucket yang sama.

3. Di akun sumber, buat kebijakan bucket untuk bucket sumber yang memberikan peran yang Anda buat untuk GET objek, ACL, tanda, dan versi di bucket objek sumber. S3 Batch Operations kemudian dapat memperoleh objek dari bucket sumber melalui peran tepercaya.

   Berikut ini adalah contoh kebijakan bucket untuk bucket yang berisi objek sumber.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowBatchOperationsSourceObjectCOPY",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::DestinationAccountNumber:role/
BatchOperationsDestinationRoleCOPY"
      },
```

```
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetObjectAcl",
        "s3:GetObjectTagging",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionTagging"
      ],
      "Resource": "arn:aws:s3:::ObjectSourceBucket/*"
    }
  ]
}
```

4. Membuat pekerjaan S3 Batch Operations di akun tujuan. Anda memerlukan Amazon Resource Name (ARN) untuk peran yang Anda buat di akun tujuan.

   Untuk informasi umum tentang cara membuat pekerjaan, lihat Membuat pekerjaan Operasi Batch S3.

   Untuk informasi tentang membuat tugas menggunakan konsol, lihat Membuat pekerjaan Operasi Batch S3.

Menggunakan Operasi Batch S3 untuk mengenkripsi objek dengan Kunci Bucket S3

Pada bagian ini, Anda menggunakan operasi Salin Operasi Batch Amazon S3 untuk mengidentifikasi dan mengaktifkan enkripsi Kunci Bucket S3 pada objek yang sudah ada. Untuk informasi selengkapnya tentang Kunci Bucket S3, lihat Mengurangi biaya SSE-KMS dengan Kunci Bucket Amazon S3 dan Mengonfigurasi bucket Anda untuk menggunakan Kunci Bucket S3 dengan SSE-KMS untuk objek baru.

Topik yang dibahas dalam contoh ini mencakup hal-hal berikut:

Topik

- Prasyarat
- Langkah 1: Dapatkan daftar objek menggunakan Inventaris Amazon S3
- Langkah 2: Filter daftar objek Anda dengan S3 Select
- Langkah 3: Mengatur dan menjalankan pekerjaan Operasi Batch S3
- Ringkasan

Prasyarat

Untuk mengikuti langkah-langkah dalam prosedur ini, Anda harus memiliki Akun AWS dan setidaknya satu bucket S3 untuk menyimpan file kerja Anda dan hasil terenkripsi. Anda mungkin juga akan menemukan banyak dokumentasi Operasi Batch S3 yang berguna, termasuk topik-topik berikut:

- Dasar-dasar Operasi Batch S3
- Membuat pekerjaan Operasi Batch S3
- Operasi yang didukung oleh Operasi Batch S3
- Mengelola tugas Operasi Batch S3

Langkah 1: Dapatkan daftar objek menggunakan Inventaris Amazon S3

Untuk memulai, identifikasi bucket S3 yang berisi objek untuk mengenkripsi, dan dapatkan daftar isinya. Laporan Inventaris Amazon S3 adalah cara yang paling mudah dan terjangkau untuk melakukan hal ini. Laporan ini memberikan daftar objek dalam bucket beserta dengan metadata terkait. Bucket sumber mengacu pada bucket inventaris, dan bucket tujuan mengacu pada bucket tempat Anda menyimpan file laporan inventaris. Untuk informasi selengkapnya tentang bucket sumber dan tujuan Inventaris Amazon S3, lihat Inventaris Amazon S3.

Cara termudah untuk mengatur inventaris adalah dengan menggunakan AWS Management Console. Namun, Anda juga dapat menggunakan API REST, AWS Command Line Interface (AWS CLI), atau SDK AWS. Sebelum mengikuti langkah-langkah ini, pastikan untuk masuk ke konsol dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/. Jika Anda mengalami kesalahan izin ditolak, tambahkan kebijakan bucket ke bucket tujuan Anda. Untuk informasi selengkapnya, lihat Berikan izin untuk Inventaris S3 dan analitik S3.

Untuk mendapatkan daftar objek menggunakan Inventaris S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi, pilih Bucket, dan pilih bucket yang berisi objek yang akan dienkripsi.
3. Pada tab Manajemen, arahkan ke bagian Konfigurasi inventaris, dan pilih Buat konfigurasi inventaris.
4. Beri nama inventaris baru Anda, masukkan nama bucket S3 tujuan, dan buat prefiks tujuan secara opsional untuk Amazon S3 agar dapat menetapkan objek dalam bucket tersebut.
5. Untuk Format output, pilih CSV.

6. (Opsional) Pada bagian Bidang tambahan – opsional, pilih Enkripsi dan bidang laporan lainnya yang menarik untuk Anda. Atur frekuensi pengiriman laporan ke Harian sehingga laporan pertama akan dikirim ke bucket Anda dengan lebih cepat.
7. Pilih Buat untuk menyimpan konfigurasi Anda.

Amazon S3 membutuhkan waktu hingga 48 jam untuk memberikan laporan pertama, jadi periksa kembali saat laporan pertama tiba. Setelah menerima laporan pertama, lanjutkan ke bagian berikutnya untuk memfilter isi laporan Inventaris S3 Anda. Jika tidak ingin menerima laporan inventaris untuk bucket ini lagi, hapus konfigurasi Inventaris S3 Anda. Jika tidak, S3 akan mengirim laporan dengan jadwal harian atau mingguan.

Daftar inventaris bukanlah point-in-time tampilan tunggal dari semua objek. Daftar inventaris adalah snapshot bergulir dari item bucket, yang pada akhirnya konsisten (misalnya, daftar ini mungkin tidak berisi objek yang baru ditambahkan atau dihapus). Penggabungan Inventaris S3 dan Operasi Batch S3 akan bekerja dengan sangat baik ketika Anda bekerja dengan objek statis, atau dengan kumpulan objek yang Anda buat dua hari yang lalu atau lebih. Untuk bekerja dengan data yang lebih baru, gunakan operasi API ListObjectsV2 (GET Bucket) untuk membuat daftar objek secara manual. Bila perlu, ulangi proses untuk beberapa hari ke depan atau sampai laporan inventaris Anda menunjukkan status yang diinginkan untuk semua kunci.

Langkah 2: Filter daftar objek Anda dengan S3 Select

Setelah menerima laporan Inventaris S3, Anda dapat memfilter konten laporan agar hanya mencantumkan objek yang tidak dienkripsi dengan Kunci Bucket S3. Jika ingin semua objek bucket dienkripsi dengan Kunci Bucket S3, Anda dapat mengabaikan langkah ini. Namun, memfilter laporan Inventaris S3 Anda pada tahap ini menghemat waktu dan biaya untuk mengenkripsi ulang objek yang sebelumnya sudah Anda enkripsi.

Meskipun langkah-langkah berikut menunjukkan cara memfilter menggunakan Amazon S3 Select, Anda juga dapat menggunakan Amazon Athena. Untuk memutuskan alat yang akan digunakan, lihat file `manifest.json` laporan Inventaris S3 Anda. File ini mencantumkan jumlah file data yang terkait dengan laporan tersebut. Jika jumlahnya banyak, gunakan Amazon Athena karena dapat digunakan di beberapa objek S3, sedangkan S3 Select hanya bekerja pada satu objek dalam satu waktu. Untuk informasi selengkapnya tentang menggunakan Amazon S3 dan Athena bersama-sama, lihat Menanyakan Inventaris Amazon S3 dengan Amazon Athena dan Menggunakan Athena di posting blog Mengenkripsi objek dengan Operasi Batch Amazon S3.

Untuk memfilter laporan Inventaris S3 menggunakan S3 Select

1. Buka file `manifest.json` dari laporan inventaris Anda dan lihat bagian `fileSchema` dari JSON. Langkah ini akan memberikan informasi kueri yang Anda jalankan pada data tersebut.

   JSON berikut adalah contoh file `manifest.json` untuk inventaris berformat CSV pada bucket dengan versioning yang diaktifkan. Tergantung pada cara Anda mengonfigurasi laporan inventaris, manifes Anda mungkin akan terlihat berbeda.

```
  {
    "sourceBucket": "batchoperationsdemo",
    "destinationBucket": "arn:aws:s3:::testbucket",
    "version": "2021-05-22",
    "creationTimestamp": "1558656000000",
    "fileFormat": "CSV",
    "fileSchema": "Bucket, Key, VersionId, IsLatest, IsDeleteMarker,
 BucketKeyStatus",
    "files": [
      {
        "key": "demoinv/batchoperationsdemo/DemoInventory/data/009a40e4-
f053-4c16-8c75-6100f8892202.csv.gz",
        "size": 72691,
        "MD5checksum": "c24c831717a099f0ebe4a9d1c5d3935c"
      }
    ]
  }
```

   Jika versioning tidak diaktifkan pada bucket, atau jika Anda memilih untuk menjalankan laporan untuk versi terbaru, `fileSchema` adalah `Bucket`, `Key`, dan `BucketKeyStatus`.

   Jika versioning diaktifkan, tergantung pada cara Anda mengatur laporan inventaris, `fileSchema` mungkin berisi kolom berikut: `Bucket`, `Key`, `VersionId`, `IsLatest`, `IsDeleteMarker`, `BucketKeyStatus`. Jadi, perhatikan kolom 1, 2, 3, dan 6 ketika Anda menjalankan kueri.

   Operasi Batch S3 membutuhkan bucket, kunci, dan ID versi sebagai input untuk menjalankan pekerjaan, selain bidang yang akan dicari, yaitu `BucketKeyStatus`. Anda tidak memerlukan bidang ID versi, tetapi bidang ini akan sangat membantu jika Anda menentukannya ketika beroperasi pada bucket berversi. Untuk informasi selengkapnya, lihat Bekerja dengan objek di dalam bucket dengan dukungan Penentuan Versi.

2. Temukan file data untuk laporan inventaris. Objek `manifest.json` mencantumkan file data di bawah file.
3. Setelah Anda menemukan dan memilih file data di konsol S3, pilih Tindakan, lalu pilih Kueri dengan S3 Select.
4. Simpan preset bidang CSV, Koma, dan GZIP yang dipilih, lalu pilih Selanjutnya.
5. Untuk meninjau format laporan inventaris Anda sebelum melanjutkan, pilih Tampilkan pratinjau file.
6. Masukkan kolom untuk referensi di bidang ekspresi SQL, dan pilih Jalankan SQL. Ekspresi berikut mengembalikan kolom 1–3 untuk semua objek tanpa Kunci Bucket S3 yang dikonfigurasi.

   ```
   select s._1, s._2, s._3 from s3object s where s._6 = 'DISABLED'
   ```

   Berikut ini adalah contoh hasil.

   ```
         batchoperationsdemo,0100059%7Ethumb.jpg,lsrtIxksLu0R0ZkYPL.LhgD5caTYn6vu
         batchoperationsdemo,0100074%7Ethumb.jpg,sd2M60g6Fdazoi6D5kNARIE7KzUibmHR
         batchoperationsdemo,0100075%7Ethumb.jpg,TLYESLnl1mXD5c4BwiOIinqFrktddkoL
         batchoperationsdemo,0200147%7Ethumb.jpg,amufzfMi_fEw0Rs99rxR_HrDFlE.l3Y0
         batchoperationsdemo,0301420%7Ethumb.jpg,9qGU2SEscL.C.c_sK89trmXYIwooABSh
         batchoperationsdemo,0401524%7Ethumb.jpg,ORnEWNuB1QhHrrYAGFsZhbyvEYJ3DUor
         batchoperationsdemo,200907200065HQ
   %7Ethumb.jpg,d8LgvIVjbDR5mUVwW6pu9ahTfReyn5V4
         batchoperationsdemo,200907200076HQ
   %7Ethumb.jpg,XUT25d7.gK40u_GmnupdaZg3BVx2jN40
         batchoperationsdemo,201103190002HQ
   %7Ethumb.jpg,z.2sVRh0myqVi0BuIrngWlsRPQdb7qOS
   ```

7. Unduh hasilnya, simpan ke dalam format CSV, dan unggah ke Amazon S3 sebagai daftar objek untuk pekerjaan Operasi Batch S3.
8. Jika Anda memiliki beberapa file manifes, jalankan Kueri dengan S3 Select pada file-file tersebut. Tergantung pada ukuran hasilnya, Anda dapat menggabungkan daftar dan menjalankan satu pekerjaan Operasi Batch S3 atau menjalankan setiap daftar sebagai pekerjaan terpisah.

   Pertimbangkan harga untuk menjalankan setiap pekerjaan Operasi Batch S3 ketika Anda memutuskan jumlah pekerjaan yang akan dijalankan.

Langkah 3: Mengatur dan menjalankan pekerjaan Operasi Batch S3

Sekarang, setelah memiliki daftar CSV objek S3 yang difilter, Anda dapat memulai pekerjaan Operasi Batch S3 untuk mengenkripsi objek dengan Kunci Bucket S3.

Pekerjaan mengacu secara kolektif ke daftar (manifes) dari objek yang diberikan, operasi yang dilakukan, dan parameter yang ditentukan. Cara termudah untuk mengenkripsi kumpulan objek ini adalah dengan menggunakan operasi salin PUT dan menentukan prefiks tujuan yang sama dengan objek yang tercantum dalam manifes. Tindakan ini akan menimpa objek yang ada di dalam bucket yang tidak memiliki versi atau, dengan versioning yang diaktifkan, membuat versi yang lebih baru dan terenkripsi dari objek tersebut.

Sebagai bagian dari penyalinan objek, tentukan bahwa Amazon S3 harus mengenkripsi objek dengan enkripsi SSE-KMS dan S3. Pekerjaan ini menyalin objek, sehingga semua objek Anda akan menampilkan tanggal pembuatan yang diperbarui setelah selesai, tanpa melihat kapan Anda menambahkannya ke S3. Tentukan juga properti lain untuk kumpulan objek Anda sebagai bagian dari pekerjaan Operasi Batch S3, termasuk tanda objek dan kelas penyimpanan.

Sublangkah

- Atur kebijakan IAM Anda
- Atur peran IAM Operasi Batch Anda
- Aktifkan Kunci Bucket S3 untuk bucket yang sudah ada
- Membuat pekerjaan Operasi Batch Anda
- Jalankan pekerjaan Operasi Batch Anda
- Hal yang perlu diperhatikan

Atur kebijakan IAM Anda

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pada panel navigasi, pilih Kebijakan, lalu pilih Buat Kebijakan.
3. Pilih tab JSON. Pilih Edit kebijakan dan tambahkan contoh kebijakan IAM yang muncul di blok kode berikut.

   Setelah menyalin contoh kebijakan ke Konsol IAM Anda, ganti yang berikut ini:

   a. Ganti *SOURCE_BUCKET_FOR_COPY* dengan nama bucket sumber Anda.
   b. Ganti *DESTINATION_BUCKET_FOR_COPY* dengan nama bucket tujuan Anda.

c. Ganti *MANIFEST_KEY* dengan nama objek manifes Anda.

d. Ganti *REPORT_BUCKET* dengan nama bucket tempat Anda ingin menyimpan laporan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CopyObjectsToEncrypt",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:PutObjectTagging",
        "s3:PutObjectAcl",
        "s3:PutObjectVersionTagging",
        "s3:PutObjectVersionAcl",
        "s3:GetObject",
        "s3:GetObjectAcl",
        "s3:GetObjectTagging",
        "s3:GetObjectVersion",
        "s3:GetObjectVersionAcl",
        "s3:GetObjectVersionTagging"
      ],
      "Resource": [
        "arn:aws:s3:::SOURCE_BUCKET_FOR_COPY/*",
        "arn:aws:s3:::DESTINATION_BUCKET_FOR_COPY/*"
      ]
    },
    {
      "Sid": "ReadManifest",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": "arn:aws:s3:::MANIFEST_KEY"
    },
    {
      "Sid": "WriteReport",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject"
      ],
```

```
        "Resource": "arn:aws:s3:::REPORT_BUCKET/*"
      }
    ]
  }
```

4. Pilih Selanjutnya: Tanda.
5. Tambahkan tanda yang Anda inginkan (opsional), lalu pilih Berikutnya: Tinjau.
6. Tambahkan nama kebijakan, tambahkan deskripsi secara opsional, lalu pilih Buat kebijakan.
7. Pilih Tinjau kebijakan dan Simpan perubahan.
8. Dengan kebijakan Operasi Batch S3 yang telah selesai, konsol akan mengembalikan Anda ke halaman Kebijakan IAM. Filter nama kebijakan, pilih tombol di sebelah kiri nama kebijakan, pilih Tindakan kebijakan, lalu pilih Lampirkan.

   Untuk melampirkan kebijakan yang baru dibuat ke peran IAM, pilih pengguna, kelompok, atau peran yang sesuai di akun Anda, lalu pilih Lampirkan kebijakan. Tindakan ini akan membawa Anda kembali ke konsol IAM.

Atur peran IAM Operasi Batch Anda

1. Di Konsol IAM, di panel navigasi, pilih Peran, lalu pilih Buat peran.
2. Pilih Layanan AWS, S3, dan Operasi Batch S3. Kemudian, pilih Selanjutnya: Izin.
3. Mulai masukkan nama kebijakan IAM yang baru saja Anda buat. Pilih kotak centang di samping nama kebijakan ketika muncul, dan pilih Selanjutnya: Tanda.
4. (Opsional) Tambahkan tanda atau simpan bidang kunci dan nilai kosong untuk latihan ini. Pilih Selanjutnya: Tinjau.
5. Masukkan nama peran, lalu terima deskripsi default atau tambahkan nama peran Anda sendiri. Pilih Buat peran.
6. Pastikan bahwa pengguna yang membuat pekerjaan memiliki izin dalam contoh berikut.

   Ganti *{ACCOUNT-ID}* dengan ID Akun AWS Anda dan *{IAM_ROLE_NAME}* dengan nama yang ingin Anda tetapkan pada peran IAM yang akan Anda buat dalam langkah pembuatan pekerjaan Batch Operations nanti. Untuk informasi selengkapnya, lihat Memberikan izin untuk Operasi Batch Amazon S3.

```
            {
            "Sid": "AddIamPermissions",
```

```
            "Effect": "Allow",
            "Action": [
            "iam:GetRole",
            "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::ACCOUNT-ID:role/IAM_ROLE_NAME"
            }
```

Aktifkan Kunci Bucket S3 untuk bucket yang sudah ada

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Dalam daftar Bucket, pilih bucket yang ingin Anda aktifkan Kunci Bucket S3-nya.
3. Pilih Properti.
4. Pada Enkripsi default, pilih Edit.
5. Pada Jenis enkripsi, Anda dapat memilih antara kunci terkelola Amazon S3 (SSE-S3) dan kunci AWS Key Management Service (SSE-KMS).
6. Jika Anda memilih kunci AWS Key Management Service (SSE-KMS), pada AWS KMS key, Anda dapat menentukan kunci AWS KMS melalui salah satu opsi berikut.
   - Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari kunci AWS KMS Anda. Dari daftar kunci yang tersedia, pilih kunci KMS enkripsi simetris di Wilayah yang sama dengan bucket Anda. Baik kunci terkelola AWS (`aws/s3`) maupun kunci terkelola pelanggan Anda akan muncul dalam daftar.
   - Untuk memasukkan ARN kunci KMS, pilih Masukkan ARN kunci AWS KMS, lalu masukkan ARN kunci KMS Anda di bidang yang muncul.
   - Untuk membuat kunci terkelola pelanggan baru di konsol AWS KMS, pilih Buat kunci KMS.
7. Pada Kunci Bucket, pilih Aktifkan, lalu pilih Simpan perubahan.

Sekarang, setelah Kunci Bucket S3 diaktifkan di tingkat bucket, objek yang diunggah, diubah, atau disalin ke dalam bucket ini akan mewarisi konfigurasi enkripsi ini secara default. Hal ini mencakup objek yang disalin dengan menggunakan Operasi Batch Amazon S3.

Membuat pekerjaan Operasi Batch Anda

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi, pilih Operasi Batch, lalu pilih Buat Pekerjaan.

3. Pilih Wilayah tempat Anda menyimpan objek, lalu pilih CSV sebagai jenis manifes.
4. Masukkan jalur atau arahkan ke file manifes CSV yang telah Anda buat sebelumnya dari hasil S3 Select (atau Athena). Jika manifes Anda berisi ID versi, centang kotaknya. Pilih Selanjutnya.
5. Pilih operasi Salin, lalu pilih bucket tujuan salinan. Anda dapat membuat enkripsi sisi server tetap nonaktif. Selama tujuan bucket mengaktifkan Kunci Bucket S3, operasi penyalinan akan menerapkan Kunci Bucket S3 pada bucket tujuan.
6. (Opsional) Pilih kelas penyimpanan dan parameter lainnya sesuai keinginan. Parameter yang Anda tentukan dalam langkah ini berlaku untuk semua operasi yang dilakukan pada objek yang tercantum dalam manifes. Pilih Selanjutnya.
7. Untuk mengonfigurasi enkripsi sisi server, ikuti langkah-langkah ini:

   a. Pada Enkripsi sisi server, pilih salah satu langkah berikut:

      - Untuk mempertahankan pengaturan bucket untuk enkripsi sisi server default pada objek saat menyimpannya di Amazon S3, pilih Jangan tentukan kunci enkripsi. Selama tujuan bucket telah mengaktifkan Kunci Bucket S3, operasi penyalinan akan menerapkan Kunci Bucket S3 di bucket tujuan.

        > **Note**
        >
        > Jika kebijakan bucket untuk tujuan yang telah ditentukan mengharuskan objek dienkripsi sebelum menyimpannya di Amazon S3, Anda harus menentukan kunci enkripsi. Jika tidak, proses penyalinan objek ke tujuan akan gagal.

      - Untuk mengenkripsi objek sebelum menyimpannya di Amazon S3, pilih Tentukan kunci enkripsi.

   b. Pada Pengaturan enkripsi, jika Anda memilih Tentukan kunci enkripsi, Anda harus memilih Gunakan pengaturan bucket tujuan untuk enkripsi default atau Ganti pengaturan bucket tujuan untuk enkripsi default.

   c. Jika memilih Ganti pengaturan bucket tujuan untuk enkripsi default, Anda harus mengonfigurasi pengaturan enkripsi berikut.

      i. Pada Jenis enkripsi, Anda harus memilih kunci terkelola Amazon S3 (SSE-S3) atau kunci AWS Key Management Service (SSE-KMS). SSE-S3 menggunakan salah satu penyandian blok terkuat—Advanced Encryption Standard 256-bit (AES-256) untuk mengenkripsi setiap objek. SSE-KMS memberikan kontrol yang lebih besar terhadap kunci Anda. Lihat informasi yang lebih lengkap di Menggunakan enkripsi di sisi server

dengan kunci terkelola Amazon S3 (SSE-S3) dan Menggunakan enkripsi sisi server dengan AWS KMS kunci (SSE-KMS).

ii. Jika Anda memilih kunci AWS Key Management Service (SSE-KMS), pada AWS KMS key, Anda dapat menentukan AWS KMS key melalui salah satu opsi berikut.

- Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari AWS KMS keys Anda, lalu pilih kunci KMS enkripsi simetris di Wilayah yang sama dengan bucket Anda. Baik kunci terkelola AWS (`aws/s3`) maupun kunci terkelola pelanggan Anda akan muncul dalam daftar.
- Untuk memasukkan ARN kunci KMS, pilih Masukkan ARN kunci AWS KMS, lalu masukkan ARN kunci KMS Anda di bidang yang muncul.
- Untuk membuat kunci terkelola pelanggan baru di konsol AWS KMS, pilih Buat kunci KMS.

iii. Pada Kunci Bucket, pilih Aktifkan. Operasi penyalinan akan menerapkan Kunci Bucket S3 pada bucket tujuan.

8. Berikan deskripsi pekerjaan Anda (atau tetap menggunakan default), atur tingkat prioritasnya, pilih jenis laporan, dan tentukan Jalur ke tujuan laporan penyelesaian.
9. Pada bagian Izin, pastikan untuk memilih peran IAM Operasi Batch yang telah Anda tentukan sebelumnya. Pilih Berikutnya.
10. Pada Peninjauan, verifikasikan pengaturan. Jika Anda ingin membuat perubahan, pilih Sebelumnya. Setelah mengonfirmasi pengaturan Operasi Batch, pilih Buat pekerjaan.

    Untuk informasi selengkapnya, lihat Membuat pekerjaan Operasi Batch S3.

Jalankan pekerjaan Operasi Batch Anda

Wizard penyiapan secara otomatis mengembalikan Anda ke bagian Operasi Batch S3 dari konsol Amazon S3. Status pekerjaan baru Anda akan berubah dari Baru ke Mempersiapkan saat S3 memulai proses. Selama status Mempersiapkan, S3 membaca manifes pekerjaan, memeriksa adanya kesalahan, dan menghitung jumlah objek.

1. Pilih tombol penyegaran di konsol Amazon S3 untuk memeriksa kemajuan. Tergantung pada ukuran manifes, pembacaan bisa memerlukan waktu beberapa menit atau bahkan jam.
2. Setelah S3 selesai membaca manifes pekerjaan, pekerjaan tersebut akan berubah ke status Menunggu konfirmasi Anda. Pilih tombol opsi di sebelah kiri ID Pekerjaan, dan pilih Jalankan pekerjaan.

3. Periksa pengaturan pekerjaan, lalu pilih Jalankan pekerjaan di pojok kanan bawah.

   Setelah pekerjaan mulai berjalan, Anda dapat memilih tombol penyegaran untuk memeriksa kemajuan melalui tampilan dasbor konsol atau dengan memilih pekerjaan tertentu.

4. Setelah pekerjaan selesai, Anda dapat melihat jumlah objek yang Berhasil dan yang Gagal untuk mengonfirmasi bahwa semua hasilnya sesuai harapan. Jika Anda mengaktifkan laporan pekerjaan, periksa laporan pekerjaan untuk mengetahui penyebab pasti dari operasi yang gagal.

   Anda juga dapat melakukan langkah-langkah ini dengan menggunakan AWS CLI, SDK AWS, atau API REST Amazon S3. Untuk informasi selengkapnya tentang melacak status pekerjaan dan laporan penyelesaian, lihat Melacak status tugas dan laporan penyelesaian.

Hal yang perlu diperhatikan

Pertimbangkan masalah berikut ketika Anda menggunakan Operasi Batch S3 untuk mengenkripsi objek dengan Kunci Bucket S3:

- Anda akan dikenakan biaya untuk pekerjaan, objek, dan permintaan Operasi Batch S3 selain biaya lain yang terkait dengan operasi yang dilakukan Operasi Batch S3 atas nama Anda, termasuk transfer data, permintaan, dan biaya lainnya. Untuk informasi selengkapnya, lihat Harga Amazon S3.
- Jika Anda menggunakan bucket yang memiliki versi, setiap pekerjaan Operasi Batch S3 yang dilakukan akan membuat versi terenkripsi baru dari objek Anda. Pekerjaan ini juga mempertahankan versi sebelumnya tanpa Kunci Bucket S3 yang dikonfigurasi. Untuk menghapus versi lama, atur kebijakan kedaluwarsa Siklus Hidup S3 untuk versi yang tidak lagi digunakan seperti yang dijelaskan di Elemen konfigurasi Siklus Hidup.
- Operasi salin menciptakan objek baru dengan tanggal pembuatan baru, yang dapat mempengaruhi tindakan siklus hidup seperti pengarsipan. Jika Anda menyalin semua objek dalam bucket, semua salinan baru memiliki tanggal pembuatan yang sama atau serupa. Untuk mengidentifikasi objek ini lebih lanjut dan membuat aturan siklus hidup yang berbeda untuk berbagai subset data, gunakan tanda objek.

Ringkasan

Di bagian ini, Anda mengurutkan objek yang ada untuk memfilter data yang sudah dienkripsi. Kemudian, Anda menerapkan fitur Kunci Bucket S3 pada objek yang tidak terenkripsi dengan menggunakan Operasi Batch S3 untuk menyalin data yang ada ke bucket dengan Kunci Bucket

S3 yang aktif. Proses ini dapat menghemat waktu dan biaya serta memungkinkan Anda untuk menyelesaikan operasi seperti mengenkripsi semua objek yang ada.

Untuk informasi selengkapnya tentang Operasi Batch S3, lihat Melakukan operasi batch berskala besar pada objek Amazon S3.

Untuk contoh yang menunjukkan operasi penyalinan dengan tanda menggunakan AWS CLI dan AWS SDK for Java, lihat Membuat tugas Operasi Batch dengan tanda pekerjaan yang digunakan untuk pelabelan.

Memanggil fungsi AWS Lambda

Fungsi Invoke memulai AWS Lambda AWS Lambda fungsi untuk melakukan tindakan kustom pada objek yang terdaftar dalam manifes. Bagian ini menjelaskan cara membuat fungsi Lambda yang akan digunakan dengan Operasi Batch S3 dan cara membuat tugas untuk menginvokasi fungsi tersebut. Tugas Operasi Batch S3 menggunakan operasi `LambdaInvoke` untuk menjalankan fungsi Lambda di setiap objek yang tercantum dalam manifes.

Anda dapat bekerja dengan Operasi Batch S3 untuk Lambda menggunakan AWS Management Console AWS Command Line Interface ,AWS CLI(), SDK AWS , atau REST API. Untuk informasi selengkapnya tentang menggunakan Lambda, lihat Memulai AWS Lambda di AWS Lambda Panduan Developer.

Bagian berikut menjelaskan cara mulai menggunakan Operasi Batch S3 dengan Lambda.

Topik

- Menggunakan Lambda dengan operasi batch Amazon S3
- Membuat fungsi Lambda untuk digunakan dengan Operasi Batch S3
- Membuat tugas Operasi Batch S3 yang menginvokasi fungsi Lambda
- Memberikan informasi tingkat tugas dalam manifest Lambda
- Belajar dari tutorial Operasi Batch S3

Menggunakan Lambda dengan operasi batch Amazon S3

Saat menggunakan Operasi Batch S3 dengan AWS Lambda, Anda harus membuat fungsi Lambda baru khusus untuk digunakan dengan Operasi Batch S3. Anda tidak dapat menggunakan kembali fungsi berbasis peristiwa Amazon S3 yang ada dengan Operasi Batch S3. Fungsi peristiwa hanya dapat menerima pesan; fungsi tersebut tidak dapat mengembalikan pesan. Fungsi Lambda yang

digunakan dengan Operasi Batch S3 harus menerima dan mengembalikan pesan. Untuk informasi selengkapnya tentang penggunaan Lambda dengan peristiwa Amazon S3, lihat Menggunakan dengan AWS Lambda Amazon S3 di Panduan Pengembang.AWS Lambda

Buat tugas Operasi Batch S3 yang menginvokasi fungsi Lambda. Tugas ini menjalankan fungsi Lambda yang sama pada semua objek yang tercantum dalam manifes Anda. Anda dapat mengontrol versi fungsi Lambda yang akan digunakan saat memproses objek dalam manifes Anda. Operasi Batch S3 mendukung Amazon Resource Name (ARN), alias, dan versi tertentu yang tidak memenuhi syarat. Untuk informasi selengkapnya, lihat Pengenalan AWS Lambda Penentuan Versi di AWS Lambda Panduan Developer.

Jika Anda menyediakan tugas Operasi Batch S3 dengan fungsi ARN yang menggunakan alias atau pengkualifikasi `$LATEST`, dan Anda memperbarui versi yang ditunjuk oleh salah satu dari keduanya, Operasi Batch S3 akan mulai memanggil versi baru fungsi Lambda. Tindakan ini sangat berguna apabila Anda ingin memperbarui fungsionalitas di tengah-tengah pekerjaan yang besar. Jika Anda tidak ingin Operasi Batch S3 mengubah versi yang digunakan, sediakan versi khusus di parameter `FunctionARN` saat Anda membuat tugas.

Menggunakan Lambda dan operasi batch Amazon S3 dengan bucket direktori

Bucket direktori adalah jenis bucket Amazon S3 yang dirancang untuk beban kerja atau aplikasi kritis kinerja yang memerlukan latensi milidetik satu digit yang konsisten. Untuk informasi selengkapnya, lihat Bucket direktori.

Ada persyaratan khusus untuk menggunakan operasi batch Amazon S3 untuk menginvokasi fungsi Lambda yang bekerja pada bucket direktori. Misalnya, Anda harus menyusun permintaan Lambda menggunakan skema JSON yang diperbarui, dan menentukan `InvocationSchemaVersion` **`2.0`** kapan Anda membuat tugas. Skema yang diperbarui ini memungkinkan Anda menentukan pasangan nilai kunci opsional untuk `UserArguments`, yang dapat Anda gunakan untuk memodifikasi parameter tertentu dari fungsi Lambda yang ada. Untuk informasi selengkapnya, lihat Mengotomatiskan pemrosesan objek di bucket direktori Amazon S3 dengan Operasi Batch S3 AWS Lambda dan di Blog Penyimpanan.AWS

Kode respons dan hasil

Operasi Batch S3 memanggil fungsi Lambda dengan satu atau lebih tombol, yang masing-masing memiliki yang `TaskID` terkait dengannya. Operasi Batch S3 mengharapkan kode hasil per kunci dari fungsi Lambda. Setiap ID tugas yang dikirim dalam permintaan yang tidak dikembalikan dengan kode hasil per-kunci akan diberikan kode hasil dari bidang. `treatMissingKeysAs` `treatMissingKeysAs`adalah bidang permintaan opsional dan default ke. `TemporaryFailure`

Tabel berikut berisi kode hasil dan nilai lain yang mungkin untuk `treatMissingKeysAs` bidang tersebut.

Kode respons	Deskripsi
`Succeeded`	Tugas diselesaikan secara normal. Jika Anda meminta laporan penyelesaian tugas, string hasil tugas disertakan dalam laporan.
`TemporaryFailure`	Tugas mengalami kegagalan sementara dan akan diarahkan ulang sebelum pekerjaan selesai. String hasil diabaikan. Jika ini adalah pengarahan ulang terakhir, pesan kesalahan akan disertakan dalam laporan akhir.
`PermanentFailure`	Tugas tersebut mengalami kegagalan permanen. Jika Anda meminta laporan penyelesaian tugas, tugas tersebut akan ditandai sebagai `Failed` dan menyertakan string pesan kesalahan. String hasil dari tugas yang gagal diabaikan.

Membuat fungsi Lambda untuk digunakan dengan Operasi Batch S3

Bagian ini memberikan contoh izin AWS Identity and Access Management (IAM) yang harus Anda gunakan dengan fungsi Lambda Anda. Bagian juga berisi contoh fungsi Lambda untuk digunakan dengan Operasi Batch S3. Jika Anda belum pernah membuat fungsi Lambda sebelumnya, lihat Tutorial: Menggunakan AWS Lambda dengan Amazon S3 di AWS Lambda Panduan Pengembang.

Anda harus membuat fungsi Lambda khusus yang akan digunakan dengan Operasi Batch S3. Anda tidak dapat menggunakan kembali fungsi Lambda berbasis peristiwa Amazon S3 yang sudah ada. Ini dikarenakan fungsi Lambda yang digunakan untuk Operasi Batch S3 harus menerima dan mengembalikan kolom data khusus.

> **Important**
>
> AWS Lambda fungsi yang ditulis dalam Java menerima salah satu RequestHandleratau antarmuka RequestStreamHandlerhandler. Namun, untuk mendukung permintaan dan

format respons Operasi Batch S3, AWS Lambda diperlukan `RequestStreamHandler` antarmuka untuk serialisasi kustom dan deserialisasi permintaan dan respons. Antarmuka ini memungkinkan Lambda untuk meneruskan InputStream dan OutputStream ke metode Java`handleRequest`.
Pastikan untuk menggunakan antarmuka `RequestStreamHandler` saat menggunakan fungsi Lambda dengan Operasi Batch S3. Jika Anda menggunakan antarmuka `RequestHandler`, tugas batch akan gagal dengan pesan "JSON yang tidak valid dikembalikan di payload Lambda" dalam laporan penyelesaian.
Untuk informasi selengkapnya, lihat Antarmuka pengelola di AWS Lambda Panduan Pengguna.

Contoh izin IAM

Berikut ini adalah contoh izin IAM yang diperlukan untuk menggunakan fungsi Lambda dengan Operasi Batch S3.

Example — Kebijakan kepercayaan Operasi Batch S3

Berikut ini adalah contoh kebijakan kepercayaan yang dapat Anda gunakan untuk peran IAM Batch Operations. Peran IAM ini ditentukan saat Anda membuat pekerjaan dan memberikan izin Operasi Batch untuk menjalankan peran IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "batchoperations.s3.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Example — Kebijakan IAM Lambda

Berikut ini adalah contoh kebijakan IAM yang memberikan izin Operasi Batch S3 untuk menginvokasi fungsi Lambda dan membaca manifes input.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "BatchOperationsLambdaPolicy",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:PutObject",
                "lambda:InvokeFunction"
            ],
            "Resource": "*"
        }
    ]
}
```

Contoh permintaan dan respons

Bagian ini memberikan contoh permintaan dan respons untuk fungsi Lambda.

Example Permintaan

Berikut ini adalah contoh JSON dari permintaan untuk fungsi Lambda.

```
{
    "invocationSchemaVersion": "1.0",
    "invocationId": "YXNkbGZqYWRmaiBhc2RmdW9hZHNmZGpmaGFzbGtkaGZza2RmaAo",
    "job": {
        "id": "f3cc4f60-61f6-4a2b-8a21-d07600c373ce"
    },
    "tasks": [
        {
            "taskId": "dGFza2lkZ29lc2hlcmUK",
            "s3Key": "customerImage1.jpg",
            "s3VersionId": "1",
            "s3BucketArn": "arn:aws:s3:us-east-1:0123456788:awsexamplebucket1"
        }
    ]
}
```

Example Respons

Berikut ini adalah contoh JSON dari respons untuk fungsi Lambda.

```
{
  "invocationSchemaVersion": "1.0",
  "treatMissingKeysAs" : "PermanentFailure",
  "invocationId" : "YXNkbGZqYWRmaiBhc2RmdW9hZHNmZGpmaGFzbGtkaGZza2RmaAo",
  "results": [
    {
      "taskId": "dGFza2lkZ29lc2hlcmUK",
      "resultCode": "Succeeded",
      "resultString": "[\"Mary Major", \"John Stiles\"]"
    }
  ]
}
```

Contoh fungsi Lambda untuk Operasi Batch S3

Contoh Python Lambda berikut menghilangkan hapus penanda dari objek berversi.

Seperti yang ditunjukkan contoh tersebut, kunci dari Operasi Batch S3 dikodekan oleh URL. Untuk menggunakan Amazon S3 dengan AWS layanan lain, penting bagi Anda untuk mendekode URL kunci yang diteruskan dari Operasi Batch S3.

```
import logging
from urllib import parse
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
logger.setLevel("INFO")

s3 = boto3.client("s3")


def lambda_handler(event, context):
    """
    Removes a delete marker from the specified versioned object.

    :param event: The S3 batch event that contains the ID of the delete marker
                  to remove.
    :param context: Context about the event.
```

```
    :return: A result structure that Amazon S3 uses to interpret the result of the
             operation. When the result code is TemporaryFailure, S3 retries the
             operation.
    """
    # Parse job parameters from Amazon S3 batch operations
    invocation_id = event["invocationId"]
    invocation_schema_version = event["invocationSchemaVersion"]

    results = []
    result_code = None
    result_string = None

    task = event["tasks"][0]
    task_id = task["taskId"]

    try:
        obj_key = parse.unquote(task["s3Key"], encoding="utf-8")
        obj_version_id = task["s3VersionId"]
        bucket_name = task["s3BucketArn"].split(":")[-1]

        logger.info(
            "Got task: remove delete marker %s from object %s.", obj_version_id,
obj_key
        )

        try:
            # If this call does not raise an error, the object version is not a delete
            # marker and should not be deleted.
            response = s3.head_object(
                Bucket=bucket_name, Key=obj_key, VersionId=obj_version_id
            )
            result_code = "PermanentFailure"
            result_string = (
                f"Object {obj_key}, ID {obj_version_id} is not " f"a delete marker."
            )

            logger.debug(response)
            logger.warning(result_string)
        except ClientError as error:
            delete_marker = error.response["ResponseMetadata"]["HTTPHeaders"].get(
                "x-amz-delete-marker", "false"
            )
            if delete_marker == "true":
                logger.info(
```

```
                    "Object %s, version %s is a delete marker.", obj_key,
obj_version_id
                )
                try:
                    s3.delete_object(
                        Bucket=bucket_name, Key=obj_key, VersionId=obj_version_id
                    )
                    result_code = "Succeeded"
                    result_string = (
                        f"Successfully removed delete marker "
                        f"{obj_version_id} from object {obj_key}."
                    )
                    logger.info(result_string)
                except ClientError as error:
                    # Mark request timeout as a temporary failure so it will be
retried.
                    if error.response["Error"]["Code"] == "RequestTimeout":
                        result_code = "TemporaryFailure"
                        result_string = (
                            f"Attempt to remove delete marker from  "
                            f"object {obj_key} timed out."
                        )
                        logger.info(result_string)
                    else:
                        raise
            else:
                raise ValueError(
                    f"The x-amz-delete-marker header is either not "
                    f"present or is not 'true'."
                )
    except Exception as error:
        # Mark all other exceptions as permanent failures.
        result_code = "PermanentFailure"
        result_string = str(error)
        logger.exception(error)
    finally:
        results.append(
            {
                "taskId": task_id,
                "resultCode": result_code,
                "resultString": result_string,
            }
        )
    return {
```

```
        "invocationSchemaVersion": invocation_schema_version,
        "treatMissingKeysAs": "PermanentFailure",
        "invocationId": invocation_id,
        "results": results,
    }
```

Membuat tugas Operasi Batch S3 yang menginvokasi fungsi Lambda

Saat membuat tugas Operasi Batch S3 untuk menginvokasi fungsi Lambda, Anda harus menyediakan hal berikut:

- ARN fungsi Lambda Anda (yang mencakup alias fungsi atau nomor versi tertentu)
- Peran IAM dengan izin untuk menginvokasi fungsi
- Parameter tindakan `LambdaInvokeFunction`

Untuk informasi selengkapnya tentang cara membuat pekerjaan Operasi Batch S3, lihat Membuat pekerjaan Operasi Batch S3 dan Operasi yang didukung oleh Operasi Batch S3.

Contoh berikut menunjukkan cara membuat tugas Operasi Batch S3 yang menginvokasi fungsi Lambda menggunakan AWS CLI.

```
aws s3control create-job
    --account-id <AccountID>
    --operation  '{"LambdaInvoke": { "FunctionArn":
 "arn:aws:lambda:Region:AccountID:function:LambdaFunctionName" } }'
    --manifest '{"Spec":{"Format":"S3BatchOperations_CSV_20180820","Fields":
["Bucket","Key"]},"Location":
{"ObjectArn":"arn:aws:s3:::ManifestLocation","ETag":"ManifestETag"}}'
    --report
 '{"Bucket":"arn:aws:s3:::awsexamplebucket1","Format":"Report_CSV_20180820","Enabled":true,"Pre
    --priority 2
    --role-arn arn:aws:iam::AccountID:role/BatchOperationsRole
    --region Region
    --description “Lambda Function"
```

Memberikan informasi tingkat tugas dalam manifest Lambda

Saat Anda menggunakan AWS Lambda fungsi dengan Operasi Batch S3, Anda mungkin ingin data tambahan menyertai setiap tugas/kunci yang dioperasikan. Misalnya, Anda ingin memiliki kunci objek sumber dan kunci objek baru yang disediakan. Fungsi Lambda Anda kemudian dapat menyalin kunci sumber ke bucket S3 baru dengan nama baru. Secara default, Operasi Batch Amazon S3 memungkinkan Anda untuk hanya menentukan bucket tujuan dan daftar kunci sumber di manifes input ke tugas Anda. Berikut ini menjelaskan cara memasukkan data tambahan ke dalam manifes sehingga Anda dapat menjalankan fungsi Lambda yang lebih kompleks.

Untuk menentukan parameter per kunci dalam manifes Operasi Batch S3 yang akan digunakan dalam kode fungsi Lambda, gunakan format JSON yang dienkode dengan URL berikut ini. Bidang `key` diteruskan ke fungsi Lambda Anda seakan-akan itu adalah kunci objek Amazon S3. Tetapi bidang tersebut dapat ditafsirkan oleh fungsi Lambda agar berisi nilai lain atau beberapa kunci, seperti yang ditunjukkan berikut ini.

> **Note**
>
> Jumlah karakter maksimum untuk bidang `key` pada manifes adalah 1.024.

Example — manifes yang menggantikan "kunci Amazon S3" dengan string JSON

Versi yang dienkode URL harus disediakan untuk Operasi Batch S3.

```
my-bucket,{"origKey": "object1key", "newKey": "newObject1Key"}
my-bucket,{"origKey": "object2key", "newKey": "newObject2Key"}
my-bucket,{"origKey": "object3key", "newKey": "newObject3Key"}
```

Example — manifes yang dienkode URL

Versi yang dienkode URL ini harus disediakan untuk Operasi Batch S3. Versi yang tidak dienkode URL tidak dapat digunakan.

```
my-bucket,%7B%22origKey%22%3A%20%22object1key%22%2C%20%22newKey%22%3A
%20%22newObject1Key%22%7D
my-bucket,%7B%22origKey%22%3A%20%22object2key%22%2C%20%22newKey%22%3A
%20%22newObject2Key%22%7D
my-bucket,%7B%22origKey%22%3A%20%22object3key%22%2C%20%22newKey%22%3A
%20%22newObject3Key%22%7D
```

Example — fungsi Lambda dengan hasil penulisan format manifes ke laporan tugas

Contoh manifes yang dikodekan URL ini berisi kunci objek yang dibatasi pipa untuk fungsi Lambda berikut untuk diuraikan.

```
my-bucket,object1key%7Clower
my-bucket,object2key%7Cupper
my-bucket,object3key%7Creverse
my-bucket,object4key%7Cdelete
```

Fungsi Lambda ini menunjukkan cara mengurai tugas yang dibatasi pipa yang dienkode ke dalam manifes Operasi Batch S3. Tugas tersebut menunjukkan operasi revisi yang diterapkan ke objek tertentu.

```
import logging
from urllib import parse
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
logger.setLevel("INFO")

s3 = boto3.resource("s3")


def lambda_handler(event, context):
    """
    Applies the specified revision to the specified object.

    :param event: The Amazon S3 batch event that contains the ID of the object to
                  revise and the revision type to apply.
    :param context: Context about the event.
    :return: A result structure that Amazon S3 uses to interpret the result of the
             operation.
    """
    # Parse job parameters from Amazon S3 batch operations
    invocation_id = event["invocationId"]
    invocation_schema_version = event["invocationSchemaVersion"]

    results = []
    result_code = None
    result_string = None
```

```
    task = event["tasks"][0]
    task_id = task["taskId"]
    # The revision type is packed with the object key as a pipe-delimited string.
    obj_key, revision = parse.unquote(task["s3Key"], encoding="utf-8").split("|")
    bucket_name = task["s3BucketArn"].split(":")[-1]

    logger.info("Got task: apply revision %s to %s.", revision, obj_key)

    try:
        stanza_obj = s3.Bucket(bucket_name).Object(obj_key)
        stanza = stanza_obj.get()["Body"].read().decode("utf-8")
        if revision == "lower":
            stanza = stanza.lower()
        elif revision == "upper":
            stanza = stanza.upper()
        elif revision == "reverse":
            stanza = stanza[::-1]
        elif revision == "delete":
            pass
        else:
            raise TypeError(f"Can't handle revision type '{revision}'.")

        if revision == "delete":
            stanza_obj.delete()
            result_string = f"Deleted stanza {stanza_obj.key}."
        else:
            stanza_obj.put(Body=bytes(stanza, "utf-8"))
            result_string = (
                f"Applied revision type '{revision}' to " f"stanza {stanza_obj.key}."
            )

        logger.info(result_string)
        result_code = "Succeeded"
    except ClientError as error:
        if error.response["Error"]["Code"] == "NoSuchKey":
            result_code = "Succeeded"
            result_string = (
                f"Stanza {obj_key} not found, assuming it was deleted "
                f"in an earlier revision."
            )
            logger.info(result_string)
        else:
            result_code = "PermanentFailure"
            result_string = (
```

```
                f"Got exception when applying revision type '{revision}' "
                f"to {obj_key}: {error}."
            )
            logger.exception(result_string)
    finally:
        results.append(
            {
                "taskId": task_id,
                "resultCode": result_code,
                "resultString": result_string,
            }
        )
    return {
        "invocationSchemaVersion": invocation_schema_version,
        "treatMissingKeysAs": "PermanentFailure",
        "invocationId": invocation_id,
        "results": results,
    }
```

Belajar dari tutorial Operasi Batch S3

Tutorial berikut menyajikan end-to-end prosedur lengkap untuk beberapa tugas Operasi Batch dengan Lambda.

- Tutorial: Video transcoding batch dengan Operasi Batch S3,, dan AWS LambdaAWS Elemental MediaConvert

Ganti semua tanda objek

Operasi Ganti semua tanda objek menggantikan tanda objek Amazon S3 pada setiap objek yang tercantum dalam manifes. Tag objek Amazon S3 adalah pasangan string dengan nilai kunci yang dapat Anda gunakan untuk menyimpan metadata tentang suatu objek.

Untuk membuat tugas Ganti semua tanda objek, Anda menyediakan sekumpulan tandag yang ingin Anda terapkan. Operasi Batch S3 menerapkan kumpulan tanda yang sama untuk setiap objek. Rangkaian tag yang Anda berikan menggantikan rangkaian tag apa pun yang sudah terkait dengan

objek dalam manifest. Operasi Batch S3 tidak mendukung penambahan tanda ke objek sambil membiarkan tanda yang ada di tempatnya.

Jika objek dalam manifes Anda berada dalam bucket berversi, Anda dapat menerapkan kumpulan tanda ke versi tertentu dari setiap objek. Anda melakukan ini dengan menentukan ID versi untuk setiap objek dalam manifes. Jika Anda tidak menyertakan ID versi untuk objek apa pun, maka Operasi Batch S3 menerapkan tanda yang diatur ke versi terbaru dari setiap objek.

Pembatasan dan batasan

- Peran (IAM) AWS Identity and Access Management yang Anda tentukan untuk menjalankan pekerjaan Operasi Batch harus memiliki izin untuk melakukan operasi Batch Amazon S3 Ganti semua objek tanda. Untuk informasi selengkapnya tentang izin yang diperlukan, lihat PutObjectTagging di Referensi API Amazon Simple Storage Service.
- Operasi Batch S3 menggunakan operasi Amazon S3 PutObjectTagging untuk menerapkan tanda ke setiap objek dalam manifes. Semua pembatasan dan batasan yang berlaku untuk operasi yang mendasarinya juga berlaku untuk pekerjaan Operasi Batch S3.

Untuk informasi selengkapnya tentang menggunakan konsol untuk membuat tugas, lihat Membuat tugas operasi batch S3.

Untuk informasi lebih lanjut tentang pemberian tanda objek, lihat Mengategorikan penyimpanan Anda menggunakan tag dalam panduan ini, dan lihat PutObjectTagging, GetObjectTagging, dan deleteObjectTagging dalam Referensi API Amazon Simple Storage Service.

Hapus semua tanda objek

Operasi Hapus semua tanda objek menghapus semua tanda objek Amazon S3 set saat ini terkait dengan objek yang tercantum dalam manifes. Operasi Batch S3 tidak mendukung menghapus tanda dari objek sekaligus menjaga tanda lain di tempatnya.

Jika objek dalam manifes Anda berada dalam bucket berversi, Anda dapat menghapus kumpulan tanda dari versi objek tertentu. Lakukan ini dengan menentukan ID versi untuk setiap objek dalam manifes. Jika Anda tidak menyertakan ID versi untuk suatu objek, Operasi Batch S3 menghapus kumpulan tanda dari versi terbaru setiap objek.

Untuk informasi lebih lanjut tentang manifes Operasi Batch, lihat Menentukan manifes.

> **Warning**
>
> Menjalankan tugas ini akan menghapus semua kumpulan tanda objek pada setiap objek yang tercantum dalam manifes.

Pembatasan dan batasan

- Peran (IAM) AWS Identity and Access Management yang Anda tentukan untuk menjalankan pekerjaan harus memiliki izin untuk melakukan operasi penandaan objek Amazon S3 Delete yang mendasarinya. Untuk informasi selengkapnya, lihat DeleteObjectTagging di Referensi API Amazon Simple Storage Service.
- Operasi Batch S3 menggunakan operasi Amazon S3 deleteObjectTaggingununtuk menghapus set tanda dari setiap objek dalam manifes. Semua batasan dan batasan yang berlaku untuk operasi yang mendasarinya juga berlaku untuk tugas Operasi Batch S3.

Untuk informasi selengkapnya tentang cara membuat tugas, lihat Membuat pekerjaan Operasi Batch S3.

Untuk detail selengkapnya tentang penandaan objek, lihat Ganti semua tanda objek dalam panduan ini, dan PutObjectTagging, GetObjectTagging, dan DeleteObjectTagging di Referensi API Amazon Simple Storage Service.

Mengganti daftar kontrol akses

Operasi Mengganti daftar kontrol akses (ACL) mengganti daftar kontrol akses Amazon S3 (ACL) untuk setiap objek yang tercantum dalam manifes. Menggunakan ACL, Anda dapat menentukan siapa yang dapat mengakses objek dan tindakan apa yang dapat mereka lakukan.

Operasi Batch S3 mendukung ACL kustom yang Anda tetapkan dan ACL kalengan yang disediakan Amazon S3 dengan seperangkat izin akses yang telah ditentukan sebelumnya.

Jika objek dalam manifes Anda berada dalam bucket berversi, Anda dapat menerapkan ACL ke versi tertentu dari setiap objek. Anda melakukan ini dengan menentukan ID versi untuk setiap objek dalam manifes. Jika Anda tidak menyertakan ID versi untuk objek apa pun, maka S3 Batch Operations akan menerapkan ACL ke versi terbaru objek tersebut.

Untuk informasi selengkapnya tentang ACL di Amazon S3, Gambaran umum daftar kontrol akses (ACL).

Blokir Akses Publik S3

Jika Anda ingin membatasi akses publik ke semua objek dalam bucket, Anda harus menggunakan Amazon S3 Block Public Access alih-alih Operasi Batch S3. Blokir Akses Publik dapat membatasi akses publik pada basis per-bucket atau seluruh akun dengan satu operasi sederhana yang berlaku dengan cepat. Hal ini menjadikannya pilihan yang lebih baik jika tujuan Anda adalah mengontrol akses publik ke semua objek dalam bucket atau akun. Gunakan Operasi Batch S3 saat Anda perlu menerapkan ACL yang disesuaikan ke setiap objek dalam manifes. Untuk informasi lebih lanjut tentang Akses Publik Blok S3, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

S3 Object Ownership

Jika objek dalam manifes berada dalam bucket yang menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek, operasi Replace access control list (ACL) hanya dapat menentukan ACL objek yang memberikan kontrol penuh kepada pemilik bucket. Operasi tidak dapat memberikan izin ACL objek ke grup lainAkun AWS atau. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Pembatasan dan batasan

- Peran yang Anda tentukan untuk menjalankan tugas Ganti daftar kontrol akses harus memiliki izin untuk melakukan operasi `PutObjectAcl` Amazon S3 yang mendasarinya. Untuk informasi lebih lanjut tentang izin yang diperlukan, lihat PutObjectAcldi Referensi API Amazon Simple Storage Service.
- Operasi Batch S3 menggunakan operasi `PutObjectAcl` Amazon S3 untuk menerapkan ACL yang ditentukan ke setiap objek dalam manifes. Oleh karena itu, semua pembatasan dan batasan yang berlaku pada PutObjectAcl operasi yang mendasarinya juga berlaku untuk tugas Operasi Batch S3 Ganti daftar kontrol akses.

Memulihkan objek dengan Operasi Batch

Operasi Pemulihan memulai permintaan pemulihan untuk objek Amazon S3 yang diarsipkan yang tercantum dalam manifes Anda. Objek yang diarsipkan berikut ini harus dipulihkan sebelum dapat diakses secara real time:

- Objek yang diarsipkan di kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive
- Objek yang diarsipkan melalui kelas penyimpanan S3 Intelligent-Tiering di tingkat Archive Access atau Deep Archive Access

Menggunakan operasi S3 Initiate Restore Object dalam pekerjaan Operasi Batch S3 Anda akan menghasilkan permintaan pemulihan untuk setiap objek yang ditentukan dalam manifes.

> **Important**
>
> Pekerjaan S3 Initiate Restore Object hanya memulai permintaan untuk memulihkan objek. Operasi Batch S3 melaporkan pekerjaan sebagai selesai untuk setiap objek setelah permintaan dimulai untuk objek tersebut. Amazon S3 tidak memperbarui pekerjaan atau mengirim notifikasi saat objek telah dipulihkan. Namun, Anda dapat menggunakan Notifikasi Peristiwa S3 untuk menerima notifikasi ketika objek tersebut tersedia di Amazon S3. Untuk informasi selengkapnya, lihat Notifikasi Peristiwa Amazon S3.

Saat Anda membuat pekerjaan S3 Initiate Restore Object, argumen berikut tersedia:

ExpirationInDays

Argumen ini menentukan berapa lama objek S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive tetap tersedia di Amazon S3. Memulai pekerjaan Pemulihan Object yang menargetkan objek S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive mengharuskan Anda mengatur `ExpirationInDays` ke 1 atau lebih.

> **Important**
>
> Jangan atur `ExpirationInDays` saat membuat pekerjaan operasi S3 Initiate Restore Object yang menargetkan objek tingkat Archive Access dan Deep Archive Access S3 Intelligent-Tiering. Objek dalam tingkat akses arsip S3 Intelligent-Tiering tidak dibatasi oleh kedaluwarsa pemulihan, sehingga menentukan hasil `ExpirationInDays` akan mengakibatkan kegagalan permintaan pemulihan.

GlacierJobTier

Amazon S3 dapat memulihkan objek menggunakan salah satu dari tiga tingkat pengambilan yang berbeda: `EXPEDITED`, `STANDARD`, dan `BULK`. Namun, fitur Operasi Batch S3 hanya mendukung tingkatan `STANDARD` pengambilan. Untuk informasi selengkapnya tentang perbedaan antar tingkat pengambilan, lihat Opsi pengambilan arsip.

Untuk informasi selengkapnya tentang harga untuk setiap tingkat, lihat bagian Permintaan & pengambilan data di halaman harga Amazon S3.

Perbedaan saat memulihkan dari S3 Glacier dan S3 Intelligent-Tiering

Memulihkan file yang diarsipkan dari kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive berbeda dengan memulihkan file dari kelas penyimpanan S3 Intelligent-Tiering di tingkat Archive Access atau Deep Archive Access.

- Saat Anda memulihkan dari S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, salinan sementara dari objek akan dibuat. Amazon S3 menghapus salinan ini setelah nilai yang Anda tentukan dalam argumen `ExpirationInDays` telah terlewati. Setelah salinan sementara ini dihapus, Anda harus mengirimkan permintaan pemulihan tambahan untuk mengakses objek.
- Saat memulihkan objek S3 Intelligent-Tiering yang diarsipkan, jangan tentukan argumen `ExpirationInDays`. Saat Anda memulihkan objek dari tingkat Archive Access atau Deep Archive Access S3 Intelligent-Tiering, objek dipindah kembali ke tingkat Frequent Access S3 Intelligent-Tiering. Setelah minimal 90 hari berturut-turut tanpa akses, objek secara otomatis dipindah ke tingkat Archive Access. Setelah minimal 180 hari berturut-turut tanpa akses, objek secara otomatis dipindah ke tingkat Deep Archive Access.
- Tugas Operasi Batch dapat beroperasi baik pada objek kelas penyimpanan S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive atau di objek tingkatan penyimpanan S3 Intelligent-Tiering Archive Access dan Deep Archive Access. Operasi Batch tidak dapat beroperasi pada kedua jenis objek yang diarsipkan dalam tugas yang sama. Untuk memulihkan kedua jenis objek tersebut, Anda harus membuat pekerjaan Operasi Batch terpisah.

Pemulihan yang tumpang tindih

Jika pekerjaan S3 Initiate Restore Object Anda mencoba memulihkan objek yang sudah dalam proses pemulihan, Operasi Batch S3 akan berjalan sebagai berikut.

Operasi pemulihan objek berhasil jika salah satu dari kondisi berikut ini benar:

- Dibandingkan dengan permintaan pemulihan yang sedang berlangsung, nilai `ExpirationInDays` pekerjaan ini sama dan nilai `GlacierJobTier` lebih cepat.
- Permintaan pemulihan sebelumnya telah selesai, dan objek saat ini tersedia. Dalam hal ini, Operasi Batch memperbarui tanggal kedaluwarsa objek yang dipulihkan agar sesuai dengan nilai `ExpirationInDays` yang ditentukan dalam permintaan pemulihan yang sedang berlangsung.

Operasi pemulihan objek akan gagal jika salah satu kondisi berikut ini benar:

- Permintaan pemulihan yang sedang berlangsung belum selesai, dan durasi pemulihan untuk pekerjaan ini (ditentukan berdasarkan nilai `ExpirationInDays`) berbeda dengan durasi pemulihan yang ditentukan dalam permintaan pemulihan yang sedang berlangsung.
- Tingkat pemulihan untuk pekerjaan ini (yang ditentukan berdasarkan nilai `GlacierJobTier`) sama atau lebih lambat dari tingkat pemulihan yang ditentukan dalam permintaan pemulihan yang sedang berlangsung.

Batasan

S3 Initiate Restore Object memiliki batasan sebagai berikut:

- Anda harus membuat pekerjaan di Wilayah yang sama dengan objek yang diarsipkan.
- Operasi Batch S3 tidak mendukung tingkat pengambilan `EXPEDITED`.

Untuk informasi selengkapnya tentang pemulihan objek, lihat Memulihkan objek yang diarsipkan.

Retensi Kunci Objek S3

Operasi Retensi Object Lock memungkinkan Anda untuk menerapkan tanggal penyimpanan untuk objek Anda menggunakan mode pemerintahan atau kepatuhan. Mode retensi ini menerapkan tingkat perlindungan yang berbeda. Anda dapat menerapkan salah satu mode penyimpanan ke versi objek apa pun. Tanggal penyimpanan, seperti kepemilikan legal, mencegah objek agar tidak ditimpa atau dihapus. Amazon S3 menyimpan pertahankan sampai tanggal yang ditentukan dalam metadata objek dan melindungi versi tertentu dari versi objek tersebut hingga periode penyimpanan berakhir.

Anda dapat menggunakan S3 Batch Operations dengan Object Lock untuk mengelola tanggal penyimpanan banyak objek Amazon S3 sekaligus. Anda menetapkan daftar objek target dalam manifest Anda dan mengirimkannya ke Batch Operations untuk diselesaikan. Untuk informasi lebih lanjut, lihat S3 Object Lock the section called “Periode retensi”.

Pekerjaan S3 Batch Operations Anda dengan tanggal penyimpanan akan berjalan hingga selesai, dibatalkan, atau hingga status gagal tercapai. Anda harus menggunakan S3 Batch Operations dan penyimpanan S3 Object Lock saat Anda ingin menambahkan, mengubah, atau menghapus tanggal penyimpanan untuk banyak objek dengan permintaan tunggal.

Batch Operations memverifikasi bahwa Object Lock diaktifkan di bucket Anda sebelum memproses kunci apa pun dalam manifest. Untuk melakukan operasi dan validasi, Batch Operations

membutuhkan `s3:GetBucketObjectLockConfiguration` dan izin `s3:PutObjectRetention` dalam peran IAM untuk memungkinkan Batch Operations memanggil Object Lock atas nama Anda. Untuk informasi selengkapnya, lihat the section called “Pertimbangan Kunci Objek”.

Untuk informasi tentang menggunakan operasi ini dengan REST API, lihat `S3PutObjectRetention` dalam operasi CreateJob di Referensi API Amazon Simple Storage Service.

Untuk contoh AWS Command Line Interface penggunaan operasi ini, lihat the section called “Menggunakan Operasi Batch dengan retensi Kunci Objek”. Sebagai contoh AWS SDK for Java, lihat the section called “Menggunakan Operasi Batch dengan retensi Kunci Objek”.

Pembatasan dan batasan

- S3 Batch Operations tidak membuat perubahan pada tingkat bucket apa pun.
- Versioning dan S3 Object Lock harus dikonfigurasi pada bucket tempat pekerjaan dilakukan.
- Semua objek yang tercantum dalam manifest harus berada dalam bucket yang sama.
- Operasi ini bekerja pada versi terbaru objek tersebut kecuali ada versi yang secara eksplisit ditentukan dalam manifest.
- Anda memerlukan izin `s3:PutObjectRetention` dalam peran IAM untuk menggunakannya.
- `s3:GetBucketObjectLockConfiguration`Izin IAM diperlukan untuk mengonfirmasi bahwa Object Lock diaktifkan untuk bucket S3.
- Anda hanya dapat memperpanjang periode penyimpanan objek dengan tanggal penyimpanan mode `COMPLIANCE` yang diterapkan, dan tidak dapat dipersingkat.

Pegangan hukum S3 Object Lock

Operasi Pegangan hukum Object Lock memungkinkan Anda untuk menempatkan pegangan hukum pada versi objek. Seperti mengatur periode penyimpanan, kepemilikan legal mencegah versi objek agar tidak ditimpa atau dihapus. Namun, kepemilikan legal tidak memiliki periode penyimpanan terkait dan tetap berlaku hingga dihapus.

Anda dapat menggunakan S3 Batch Operations dengan Object Lock untuk menambahkan kepemilikan legal pada banyak objek Amazon S3 sekaligus . Anda dapat melakukannya dengan mencantumkan objek target dalam manifest Anda dan mengirimkan daftar tersebut ke Batch Operations. Pekerjaan S3 Batch Operations Anda dengan kepemilikan legal Object Lock akan berjalan hingga selesai, dibatalkan, atau hingga status gagal tercapai.

S3 Batch Operations memverifikasi bahwa Object Lock diaktifkan pada bucket S3 Anda sebelum memproses kunci apa pun dalam manifest. Untuk melakukan operasi objek dan validasi level bucket, Operasi Batch S3 membutuhkan `s3:PutObjectLegalHold` dan `s3:GetBucketObjectLockConfiguration` dalam IAM role yang memungkinkan Operasi Batch S3 untuk memanggil S3 Object Lock atas nama Anda.

Saat Anda membuat pekerjaan S3 Batch Operations untuk menghapus kepemilikan legal, Anda hanya perlu menentukan Off sebagai status kepemilikan legal. Untuk informasi selengkapnya, lihat the section called “Pertimbangan Kunci Objek”.

Untuk informasi tentang cara menggunakan operasi ini dengan REST API, lihat `S3PutObjectLegalHold` dalam operasi CreateJob di Referensi API Amazon Simple Storage Service.

Untuk contoh penggunaan operasi ini, lihat Menggunakan AWS SDK for Java.

Pembatasan dan batasan

- S3 Batch Operations tidak membuat perubahan pada tingkat bucket apa pun.
- Semua objek yang tercantum dalam manifest harus berada dalam bucket yang sama.
- Versioning dan S3 Object Lock harus dikonfigurasi pada bucket tempat pekerjaan dilakukan.
- Operasi ini bekerja pada versi terbaru objek tersebut kecuali ada versi yang secara eksplisit ditentukan dalam manifest.
- `s3:PutObjectLegalHold`Izin diperlukan dalam peran IAM Anda untuk menambahkan atau menghapus kepemilikan legal dari objek.
- `s3:GetBucketObjectLockConfiguration`Izin IAM diperlukan untuk mengonfirmasi bahwa S3 Object Lock diaktifkan untuk bucket S3.

- Salin objek
- Memanggil fungsi AWS Lambda
- Ganti semua tanda objek
- Hapus semua tanda objek
- Mengganti daftar kontrol akses
- Memulihkan objek dengan Operasi Batch
- Retensi Kunci Objek S3

- Pegangan hukum S3 Object Lock
- Mereplikasi objek yang ada dengan Replikasi Batch S3

Mengelola tugas Operasi Batch S3

Amazon S3 menyediakan seperangkat alat yang andal untuk membantu mengelola tugas Operasi Batch S3 setelah dibuat. Bagian ini menjelaskan operasi yang dapat Anda gunakan untuk mengelola dan melacak tugas menggunakanAWS Management Console, AWS CLI, SDK AWS, atau API REST.

Topik

- Menggunakan konsol Amazon S3 untuk mengelola tugas Operasi Batch S3
- Membuat daftar tugas
- Melihat detail tugas
- Menetapkan prioritas tugas

Menggunakan konsol Amazon S3 untuk mengelola tugas Operasi Batch S3

Anda dapat mengelola pekerjaan Operasi Batch S3 dengan menggunakan konsol. Sebagai contoh, Anda dapat:

- Melihat pekerjaan yang aktif dan yang berada dalam antrean
- Mengubah prioritas tugas
- Mengonfirmasi dan menjalankan tugas
- Mengkloning tugas
- Membatalkan tugas

Untuk mengelola Operasi Batch menggunakan konsol

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Operasi Batch.
3. Pilih tugas spesifik yang ingin Anda kelola.

Membuat daftar tugas

Anda dapat mengambil daftar tugas Operasi Batch S3 Anda. Daftar ini mencakup tugas yang belum dan yang telah selesai dalam 90 hari terakhir. Daftar tugas mencakup informasi untuk setiap tugas, seperti ID, deskripsi, prioritas, status saat ini, dan jumlah tugas yang berhasil dan gagal. Anda dapat memfilter daftar tugas menurut statusnya. Saat mengambil daftar tugas melalui konsol, Anda juga dapat mencari tugas menurut deskripsi atau ID dan memfilternya menurut Wilayah AWS.

Mendapatkan daftar tugas yang Aktif dan Selesai

Contoh AWS CLI berikut mendapatkan daftar tugas `Active` dan `Complete`.

```
aws s3control list-jobs \
    --region us-west-2 \
    --account-id acct-id \
    --job-statuses '["Active","Complete"]' \
    --max-results 20
```

Untuk informasi dan contoh selengkapnya, lihat list-jobs di Referensi Perintah AWS CLI.

Melihat detail tugas

Untuk informasi selengkapnya mengenai suatu tugas selain dari daftar tugas, Anda dapat melihat semua detail dari setiap tugas. Anda dapat melihat detail untuk tugas yang belum atau yang telah selesai dalam 90 hari terakhir. Selain informasi yang ditampilkan dalam daftar tugas, rincian setiap tugas juga mencakup item lain, seperti:

- Parameter operasi
- Detail tentang manifes
- Informasi tentang laporan penyelesaian (jika Anda mengonfigurasinya saat membuat tugas)
- Amazon Resource Name (ARN) dari peran pengguna yang Anda tetapkan untuk menjalankan tugas

Dengan melihat detail tugas individual, Anda dapat mengakses seluruh konfigurasi tugas. Untuk melihat detail tugas, Anda dapat menggunakan konsol Amazon S3 atau AWS Command Line Interface (AWS CLI).

Mendapatkan deskripsi tugas Operasi Batch S3 di konsol Amazon S3

Untuk melihat deskripsi tugas Operasi Batch menggunakan konsol

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Operasi Batch.
3. Pilih ID tugas dari tugas tertentu untuk melihat detailnya.

Mendapatkan deskripsi tugas Operasi Batch S3 di AWS CLI

Contoh berikut mendapatkan deskripsi tugas Operasi Batch S3 menggunakan AWS CLI. Untuk menggunakan perintah contoh berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control describe-job \
--region us-west-2 \
--account-id acct-id \
--job-id 00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c
```

Untuk informasi dan contoh selengkapnya, lihat describe-job di Referensi Perintah AWS CLI.

Menetapkan prioritas tugas

Anda dapat menetapkan prioritas numerik pada setiap tugas, yang bisa berupa bilangan bulat positif. Operasi Batch S3 memprioritaskan tugas sesuai dengan prioritas yang ditetapkan. Tugas dengan prioritas yang lebih tinggi (atau nilai numerik yang lebih tinggi untuk parameter prioritas) akan dievaluasi terlebih dahulu. Prioritas ditentukan dalam urutan menurun. Misalnya, antrean tugas dengan nilai prioritas 10 diberikan preferensi penjadwalan di atas antrean tugas dengan nilai prioritas 1.

Anda dapat mengubah prioritas tugas saat tugas tersebut sedang berjalan. Jika Anda mengajukan tugas baru dengan prioritas yang lebih tinggi saat tugas sedang berjalan, tugas dengan prioritas yang lebih rendah dapat berhenti agar tugas dengan prioritas yang lebih tinggi dapat berjalan.

Mengubah prioritas tugas tidak akan mempengaruhi kecepatan pemrosesan tugas.

> **Note**
>
> Operasi Batch S3 berupaya mematuhi prioritas tugas dengan sebai-baiknya. Meskipun tugas dengan prioritas yang lebih tinggi umumnya lebih diutamakan daripada tugas dengan prioritas yang lebih rendah, Amazon S3 tidak menjamin urutan tugas secara ketat.

Menggunakan konsol S3

Cara memperbarui prioritas tugas di AWS Management Console

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Operasi Batch.
3. Pilih tugas spesifik yang ingin Anda kelola.
4. Pilih Tindakan. Dalam daftar dropdown, pilih Perbarui prioritas.

Menggunakan AWS CLI

Contoh berikut menunjukkan cara memperbarui prioritas tugas menggunakan AWS CLI. Angka yang lebih tinggi menunjukkan prioritas eksekusi yang lebih tinggi.

```
aws s3control update-job-priority \
    --region us-west-2 \
    --account-id acct-id \
    --priority 98 \
    --job-id 00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c
```

Menggunakan AWS SDK for Java

Contoh berikut memperbarui prioritas tugas Operasi Batch S3 menggunakan AWS SDK for Java.

Untuk informasi selengkapnya tentang prioritas tugas, lihat Menetapkan prioritas tugas.

Example

```
package aws.example.s3control;
```

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.UpdateJobPriorityRequest;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class UpdateJobPriority {
    public static void main(String[] args) {
        String accountId = "Account ID";
        String jobId = "00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c";

        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.updateJobPriority(new UpdateJobPriorityRequest()
                    .withAccountId(accountId)
                    .withJobId(jobId)
                    .withPriority(98));

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Melacak status tugas dan laporan penyelesaian

Dengan Operasi Batch S3, Anda dapat melihat dan memperbarui status tugas, menambahkan notifikasi dan pencatatan, melacak kegagalan tugas, dan menghasilkan laporan penyelesaian.

Topik

- Status tugas
- Memperbarui status tugas
- Notifikasi dan pencatatan
- Melacak kegagalan tugas
- Laporan penyelesaian
- Contoh: Melacak pekerjaan Operasi Batch S3 di Amazon melalui EventBridge AWS CloudTrail
- Contoh: Laporan penyelesaian Operasi Batch S3

Status tugas

Setelah Anda membuat dan menjalankan tugas, tugas tersebut akan berjalan melalui serangkaian status. Tabel berikut ini menjelaskan status dan transisi yang mungkin terjadi di antara keduanya.

Status	Deskripsi	Transisi
`New`	Tugas dimulai dengan status `New` saat Anda membuatnya.	Tugas secara otomatis akan berpindah ke status `Preparing` saat Amazon S3 mulai memproses objek manifes.
`Preparing`	Amazon S3 memproses objek manifes dan parameter tugas lainnya untuk menyiapkan dan menjalankan tugas.	Tuagas secara otomatis akan berpindah ke status `Ready` setelah Amazon S3 selesai memproses manifes dan parameter lainnya. Tugas ini kemudian siap untuk mulai menjalankan operasi tertentu pada objek yang tercantum dalam manifes. Jika tugas memerlukan konfirmasi sebelum dijalanka n, seperti saat Anda membuat tugas menggunakan konsol

Status	Deskripsi	Transisi
		Amazon S3, maka tugas tersebut akan bertransisi dari `Preparing` ke `Suspended`. Tugas akan tetap di status `Suspended` sampai Anda mengonfirmasi untuk menjalankannya.
`Suspended`	Tugas tersebut memerlukan konfirmasi, tetapi Anda belum mengonfirmasi untuk menjalankannya. Hanya tugas yang Anda buat menggunakan konsol Amazon S3 yang memerlukan konfirmasi. Tugas yang dibuat menggunakan konsol akan memasuki status `Suspended` segera setelah `Preparing`. Setelah Anda mengonfirmasi untuk menjalankan tugas dan tugas tersebut telah menjadi `Ready`, tugas itu tidak pernah kembali ke status `Suspended`.	Setelah Anda mengonfirmasi untuk menjalankan tugas, statusnya berubah menjadi `Ready`.

Status	Deskripsi	Transisi
`Ready`	Amazon S3 siap untuk mulai menjalankan pengoperasian objek yang diminta.	Tugas secara otomatis berpindah ke status `Active` saat Amazon S3 mulai menjalankannya. Lama waktu yang diperlukan tugas untuk tetap berada dalam status `Ready` tergantung pada apakah Anda sudah memiliki tugas dengan prioritas yang lebih tinggi dan berapa lama tugas tersebut harus diselesai kan.
`Active`	Amazon S3 menjalankan operasi yang diminta pada objek yang tercantum dalam manifes. Meskipun ada pekerjaan`Active`, Anda dapat memantau kemajuannya menggunakan konsol Amazon S3 atau `DescribeJob` operasi melalui REST API, AWS CLI, atau AWS SDK.	Tugas akan keluar dari status `Active` saat tidak lagi menjalankan operasi pada objek. Ini dapat terjadi secara otomatis, misalnya saat sebuah tugas berhasil atau gagal. Atau bisa juga terjadi akibat tindakan pengguna, seperti membatalkan tugas. Status perpindahan tugas tergantung pada alasan transisi.
`Pausing`	Tugas bertransisi ke `Paused` dari status lain.	Tugas secara otomatis berpindah ke `Paused` saat tahap `Pausing` selesai.

Status	Deskripsi	Transisi
`Paused`	Tugas dapat menjadi `Paused` jika Anda mengirimkan tugas lain dengan prioritas yang lebih tinggi ketika tugas saat ini sedang berjalan.	Tugas `Paused` secara otomatis kembali ke `Active` setelah tugas dengan prioritas yang lebih tinggi yang menghalangi eksekusi tugas telah selesai, gagal, atau ditangguhkan.
`Complete`	Tugas telah selesai melakukan operasi yang diminta pada semua objek dalam manifes. Operasi mungkin telah berhasil atau gagal untuk setiap objek. Jika Anda mengonfigurasi tugas untuk membuat laporan penyelesaian, laporan tersebut akan tersedia segera setelah tugas `Complete`.	`Complete` adalah status terminal. Setelah mencapai `Complete`, tugas tidak akan bertransisi ke status lain.
`Cancelling`	Tugas bertransisi ke status `Cancelled` .	Tugas secara otomatis berpindah ke `Cancelled` saat tahap `Cancelling` selesai.
`Cancelled`	Anda meminta tugas dibatalkan, dan Operasi Batch S3 telah berhasil membatalkan tugas tersebut. Tugas ini tidak akan mengirimkan permintaan baru apa pun ke Amazon S3.	`Cancelled` adalah status terminal. Setelah mencapai `Cancelled` , tugas tidak akan bertransisi ke status lain.
`Failing`	tugas beralih ke status `Failed`.	Tugas secara otomatis berpindah ke `Failed` setelah tahap `Failing` selesai.

Status	Deskripsi	Transisi
`Failed`	Tugas gagal dan tidak lagi berjalan. Untuk informasi selengkapnya tentang kegagalan tugas, lihat Melacak kegagalan tugas.	`Failed` adalah status terminal. Setelah mencapai `Failed`, tugas tidak akan bertransisi ke status lain.

Memperbarui status tugas

Contoh berikut AWS CLI dan SDK for Java memperbarui status pekerjaan Operasi Batch. Untuk informasi selengkapnya tentang menggunakan konsol S3 untuk mengelola tugas Operasi Batch, lihat Menggunakan konsol Amazon S3 untuk mengelola tugas Operasi Batch S3.

Menggunakan AWS CLI

- Jika Anda tidak menentukan parameter `--no-confirmation-required` dalam contoh `create-job` sebelumnya, tugas tetap dalam status ditangguhkan hingga Anda mengonfirmasi tugas dengan mengatur statusnya ke `Ready`. Amazon S3 kemudian membuat tugas yang memenuhi syarat untuk dijalankan.

```
aws s3control update-job-status \
    --region us-west-2 \
    --account-id 181572960644 \
    --job-id 00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c \
    --requested-job-status 'Ready'
```

- Batalkan tugas dengan mengatur status tugas menjadi `Cancelled`.

```
aws s3control update-job-status \
     --region us-west-2 \
     --account-id 181572960644 \
     --job-id 00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c \
     --status-update-reason "No longer needed" \
     --requested-job-status Cancelled
```

Menggunakan AWS SDK for Java

Contoh berikut memperbarui status tugas Operasi Batch S3 menggunakan AWS SDK for Java.

Untuk informasi selengkapnya tentang status tugas, lihat Melacak status tugas dan laporan penyelesaian.

Example

```
package aws.example.s3control;


import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.UpdateJobStatusRequest;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class UpdateJobStatus {
    public static void main(String[] args) {
        String accountId = "Account ID";
        String jobId = "00e123a4-c0d8-41f4-a0eb-b46f9ba5b07c";

        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.updateJobStatus(new UpdateJobStatusRequest()
                    .withAccountId(accountId)
                    .withJobId(jobId)
                    .withRequestedJobStatus("Ready"));

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
```

```
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Notifikasi dan pencatatan

Selain meminta laporan penyelesaian, Anda juga dapat mencatat, meninjau, dan mengaudit aktivitas Operasi Batch menggunakan AWS CloudTrail. Karena Operasi Batch menggunakan API Amazon S3 yang ada untuk menjalankan tugas, tugas tersebut juga menghasilkan peristiwa yang sama jika Anda memanggilnya secara langsung. Dengan demikian, Anda dapat melacak dan mencatat kemajuan pekerjaan dan semua tugasnya menggunakan alat dan proses notifikasi, pencatatan, dan audit yang sama dengan yang telah Anda gunakan dengan Amazon S3. Untuk informasi selengkapnya, lihat contoh di bagian berikut.

> **Note**
>
> Operasi Batch Amazon S3 menghasilkan peristiwa manajemen dan data CloudTrail selama pelaksanaan pekerjaan. Volume peristiwa ini diskalakan dengan jumlah kunci di setiap manifes tugas. Lihat halaman CloudTrail harga untuk detailnya, yang mencakup contoh perubahan harga tergantung pada jumlah jejak yang telah Anda konfigurasikan di akun Anda. Untuk mempelajari cara mengonfigurasi dan mencatat peristiwa agar sesuai dengan kebutuhan, lihat Buat jejak pertama Anda di Panduan Pengguna AWS CloudTrail .

Untuk informasi selengkapnya tentang peristiwa Amazon S3, lihat Notifikasi Peristiwa Amazon S3.

Melacak kegagalan tugas

Jika tugas Operasi Batch S3 mengalami masalah yang membuatnya mengalami kegagalan, seperti tidak dapat membaca manifes yang ditentukan, maka tugas tersebut akan gagal. Saat mengalami kegagalan, tugas akan menghasilkan satu atau beberapa kode kegagalan atau alasan kegagalan. Operasi Batch S3 menyimpan kode kegagalan dan alasannya sehingga Anda dapat memeriksanya dengan meminta detail tugas. Jika Anda meminta laporan penyelesaian untuk tugas tersebut, kode dan alasan kegagalan juga akan muncul di sana.

Agar tugas tidak menjalankan banyak operasi yang gagal, Amazon S3 memberlakukan ambang batas kegagalan tugas pada setiap tugas Operasi Batch. Setelah pekerjaan menjalankan setidaknya

1.000 tugas, Amazon S3 akan memonitor tingkat kegagalan tugas. Jika tingkat kegagalan (jumlah tugas yang gagal sebagai proporsi dari jumlah total tugas yang telah berjalan) melebihi 50 persen, maka pekerjaan tersebut gagal. Jika pekerjaan gagal karena melebihi ambang batas kegagalan tugas, Anda dapat mengidentifikasi penyebab kegagalan. Misalnya, Anda mungkin tidak sengaja memasukkan beberapa objek dalam manifes yang tidak ada dalam bucket tertentu. Setelah memperbaiki kesalahan, Anda dapat mengirim ulang tugas tersebut.

> **Note**
>
> Operasi Batch S3 beroperasi secara asinkron dan tugas-tugasnya tidak selalu berjalan sesuai dengan urutan objek yang tercantum dalam manifes. Oleh karena itu, Anda tidak dapat menggunakan urutan manifes untuk menentukan tugas objek yang berhasil maupun yang gagal. Sebagai gantinya, Anda dapat memeriksa laporan penyelesaian pekerjaan (jika Anda memintanya) atau melihat log AWS CloudTrail peristiwa Anda untuk membantu menentukan sumber kegagalan.

Laporan penyelesaian

Saat Anda membuat tugas, Anda dapat meminta laporan penyelesaian. Selama Operasi Batch S3 berhasil menginvokasi setidaknya satu tugas, Amazon S3 akan menghasilkan laporan penyelesaian setelah selesai menjalankan tugas, gagal, atau dibatalkan. Anda dapat mengonfigurasi laporan penyelesaian untuk menyertakan semua tugas atau hanya tugas yang gagal.

Laporan penyelesaian mencakup konfigurasi pekerjaan, status dan informasi untuk setiap tugas, termasuk kunci objek dan versi, status, kode kesalahan, serta deskripsi kesalahan. Laporan penyelesaian memberikan cara mudah untuk melihat hasil tugas Anda dalam format terkonsolidasi tanpa perlu pengaturan tambahan. Laporan penyelesaian dienkripsi dengan kunci terkelola Amazon S3 (SSE-S3). Untuk contoh laporan penyelesaian, lihat Contoh: Laporan penyelesaian Operasi Batch S3.

Jika Anda tidak mengonfigurasi laporan penyelesaian, Anda masih dapat memantau dan mengaudit pekerjaan Anda dan tugasnya menggunakan CloudTrail dan Amazon CloudWatch. Untuk informasi selengkapnya, lihat bagian berikut.

Topik

- Contoh: Melacak pekerjaan Operasi Batch S3 di Amazon melalui EventBridge AWS CloudTrail
- Contoh: Laporan penyelesaian Operasi Batch S3

Contoh: Melacak pekerjaan Operasi Batch S3 di Amazon melalui EventBridge AWS CloudTrail

Aktivitas tugas Operasi Batch Amazon S3 dicatat sebagai peristiwa di AWS CloudTrail. Anda dapat membuat aturan khusus di Amazon EventBridge dan mengirim peristiwa ini ke sumber daya notifikasi target pilihan Anda, seperti Amazon Simple Notification Service (Amazon SNS).

> **Note**
>
> Amazon EventBridge adalah cara terbaik untuk mengelola peristiwa Anda. Peristiwa CloudWatch Amazon dan EventBridge merupakan layanan dasar dan API yang sama, namun EventBridge menyediakan lebih banyak fitur. Perubahan yang Anda buat pada CloudWatch atau EventBridge akan muncul di setiap konsol. Untuk informasi lebih lanjut, lihat Panduan Pengguna Amazon EventBridge.

Contoh Pelacakan

- Peristiwa Operasi Batch S3 direkam dalam CloudTrail
- EventBridgeaturan untuk melacak acara pekerjaan Operasi Batch S3

Peristiwa Operasi Batch S3 direkam dalam CloudTrail

Ketika pekerjaan Batch Operations dibuat, itu dicatat sebagai `JobCreated` acara diCloudTrail. Sebagai pekerjaan berjalan, perubahan negara selama pengolahan, dan `JobStatusChanged` peristiwa lainnya dicatat dalamCloudTrail. Anda dapat melihat peristiwa ini di CloudTrailkonsol. Untuk informasi selengkapnya tentang CloudTrail, lihat Panduan Pengguna AWS CloudTrail.

> **Note**
>
> Hanya peristiwa pekerjaan S3 Batch Operations `status-change` yang dicatat dalam CloudTrail.

Example Acara penyelesaian pekerjaan S3 Batch Operations dicatat oleh CloudTrail

```
{
    "eventVersion": "1.05",
    "userIdentity": {
```

```
        "accountId": "123456789012",
        "invokedBy": "s3.amazonaws.com"
    },
    "eventTime": "2020-02-05T18:25:30Z",
    "eventSource": "s3.amazonaws.com",
    "eventName": "JobStatusChanged",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "s3.amazonaws.com",
    "userAgent": "s3.amazonaws.com",
    "requestParameters": null,
    "responseElements": null,
    "eventID": "f907577b-bf3d-4c53-b9ed-8a83a118a554",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "recipientAccountId": "123412341234",
    "serviceEventDetails": {
        "jobId": "d6e58ec4-897a-4b6d-975f-10d7f0fb63ce",
        "jobArn": "arn:aws:s3:us-west-2:181572960644:job/
d6e58ec4-897a-4b6d-975f-10d7f0fb63ce",
        "status": "Complete",
        "jobEventId": "b268784cf0a66749f1a05bce259804f5",
        "failureCodes": [],
        "statusChangeReason": []
    }
}
```

EventBridgeaturan untuk melacak acara pekerjaan Operasi Batch S3

Contoh berikut menunjukkan cara membuat aturan di Amazon EventBridge untuk menangkap peristiwa Operasi Batch S3 yang direkam AWS CloudTrail ke target pilihan Anda.

Untuk melakukan ini, Anda membuat aturan dengan mengikuti semua langkah dalam Membuat EventBridge aturan yang bereaksi terhadap peristiwa. Anda menempelkan kebijakan pola peristiwa kustom Operasi Batch S3 berikut jika berlaku, dan memilih layanan target pilihan Anda.

S3 Batch Operations kebijakan pola acara kustom

```
{
    "source": [
        "aws.s3"
    ],
    "detail-type": [
        "AWS Service Event via CloudTrail"
```

```
    ],
    "detail": {
        "eventSource": [
            "s3.amazonaws.com"
        ],
        "eventName": [
            "JobCreated",
            "JobStatusChanged"
        ]
    }
}
```

Contoh-contoh berikut adalah dua peristiwa Batch Operations yang dikirim ke Amazon Simple Queue Service (Amazon SQS) dari peraturan peristiwa EventBridge yang berbeda. Pekerjaan Batch Operations melewati berbagai status yang berbeda saat memproses (`New`, `Preparing`, `Active`, dll.), sehingga Anda dapat berharap untuk menerima beberapa pesan untuk setiap pekerjaan.

Example JobCreatedcontoh acara

```
{
    "version": "0",
    "id": "51dc8145-541c-5518-2349-56d7dffdf2d8",
    "detail-type": "AWS Service Event via CloudTrail",
    "source": "aws.s3",
    "account": "123456789012",
    "time": "2020-02-27T15:25:49Z",
    "region": "us-east-1",
    "resources": [],
    "detail": {
        "eventVersion": "1.05",
        "userIdentity": {
            "accountId": "11112223334444",
            "invokedBy": "s3.amazonaws.com"
        },
        "eventTime": "2020-02-27T15:25:49Z",
        "eventSource": "s3.amazonaws.com",
        "eventName": "JobCreated",
        "awsRegion": "us-east-1",
        "sourceIPAddress": "s3.amazonaws.com",
        "userAgent": "s3.amazonaws.com",
        "eventID": "7c38220f-f80b-4239-8b78-2ed867b7d3fa",
        "readOnly": false,
```

```
        "eventType": "AwsServiceEvent",
        "serviceEventDetails": {
            "jobId": "e849b567-5232-44be-9a0c-40988f14e80c",
            "jobArn": "arn:aws:s3:us-east-1:181572960644:job/
e849b567-5232-44be-9a0c-40988f14e80c",
            "status": "New",
            "jobEventId": "f177ff24f1f097b69768e327038f30ac",
            "failureCodes": [],
            "statusChangeReason": []
        }
    }
}
```

Example JobStatusChangedAcara Penyelesaian Pekerjaan

```
{
  "version": "0",
  "id": "c8791abf-2af8-c754-0435-fd869ce25233",
  "detail-type": "AWS Service Event via CloudTrail",
  "source": "aws.s3",
  "account": "123456789012",
  "time": "2020-02-27T15:26:42Z",
  "region": "us-east-1",
  "resources": [],
  "detail": {
    "eventVersion": "1.05",
    "userIdentity": {
      "accountId": "1111222233334444",
      "invokedBy": "s3.amazonaws.com"
    },
    "eventTime": "2020-02-27T15:26:42Z",
    "eventSource": "s3.amazonaws.com",
    "eventName": "JobStatusChanged",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "s3.amazonaws.com",
    "userAgent": "s3.amazonaws.com",
    "eventID": "0238c1f7-c2b0-440b-8dbd-1ed5e5833afb",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "serviceEventDetails": {
      "jobId": "e849b567-5232-44be-9a0c-40988f14e80c",
      "jobArn": "arn:aws:s3:us-east-1:181572960644:job/
e849b567-5232-44be-9a0c-40988f14e80c",
```

```
      "status": "Complete",
      "jobEventId": "51f5ac17dba408301d56cd1b2c8d1e9e",
      "failureCodes": [],
      "statusChangeReason": []
    }
  }
}
```

Contoh: Laporan penyelesaian Operasi Batch S3

Saat membuat pekerjaan Operasi Batch S3, Anda dapat meminta laporan penyelesaian untuk semua tugas atau hanya untuk tugas yang gagal. Asalkan setidaknya satu tugas telah berhasil diinvokasi, Operasi Batch S3 menghasilkan laporan untuk tugas yang telah selesai, gagal, atau dibatalkan.

Laporan penyelesaian berisi informasi tambahan untuk setiap pekerjaan, termasuk nama dan versi kunci objek, status, kode kesalahan, dan deskripsi kesalahan apa pun. Deskripsi kesalahan untuk setiap tugas yang gagal dapat digunakan untuk mendiagnosis masalah yang terjadi selama pembuatan tugas, misalnya izin.

> **Note**
>
> Laporan penyelesaian selalu dienkripsi dengan kunci terkelola Amazon S3 (SSE-S3).

Example file hasil manifes tingkat atas

File tingkat atas `manifest.json` berisi lokasi setiap laporan yang berhasil dan lokasi laporan yang gagal (jika tugas mengalami kegagalan), seperti yang ditunjukkan dalam contoh berikut.

```
{
    "Format": "Report_CSV_20180820",
    "ReportCreationDate": "2019-04-05T17:48:39.725Z",
    "Results": [
        {
            "TaskExecutionStatus": "succeeded",
            "Bucket": "my-job-reports",
            "MD5Checksum": "83b1c4cbe93fc893f54053697e10fd6e",
            "Key": "job-f8fb9d89-a3aa-461d-bddc-ea6a1b131955/
results/6217b0fab0de85c408b4be96aeaca9b195a7daa5.csv"
        },
        {
```

```
            "TaskExecutionStatus": "failed",
            "Bucket": "my-job-reports",
            "MD5Checksum": "22ee037f3515975f7719699e5c416eaa",
            "Key": "job-f8fb9d89-a3aa-461d-bddc-ea6a1b131955/results/
b2ddad417e94331e9f37b44f1faf8c7ed5873f2e.csv"
        }
    ],
    "ReportSchema": "Bucket, Key, VersionId, TaskStatus, ErrorCode, HTTPStatusCode,
 ResultMessage"
}
```

Example laporan tugas yang gagal

Laporan tugas yang gagal berisi informasi berikut untuk semua tugas yang gagal:

- `Bucket`
- `Key`
- `VersionId`
- `TaskStatus`
- `ErrorCode`
- `HTTPStatusCode`
- `ResultMessage`

Contoh laporan berikut menunjukkan kasus di mana AWS Lambda fungsi habis waktu, menyebabkan kegagalan melebihi ambang kegagalan. Kemudian ditandai sebagai `PermanentFailure`.

```
awsexamplebucket1,image_14975,,failed,200,PermanentFailure,"Lambda returned
 function error: {""errorMessage"":""2019-04-05T17:35:21.155Z 2845ca0d-38d9-4c4b-
abcf-379dc749c452 Task timed out after 3.00 seconds""}"
awsexamplebucket1,image_15897,,failed,200,PermanentFailure,"Lambda returned
 function error: {""errorMessage"":""2019-04-05T17:35:29.610Z 2d0a330b-de9b-425f-
b511-29232fde5fe4 Task timed out after 3.00 seconds""}"
awsexamplebucket1,image_14819,,failed,200,PermanentFailure,"Lambda returned function
 error: {""errorMessage"":""2019-04-05T17:35:22.362Z fcf5efde-74d4-4e6d-b37a-
c7f18827f551 Task timed out after 3.00 seconds""}"
awsexamplebucket1,image_15930,,failed,200,PermanentFailure,"Lambda returned function
 error: {""errorMessage"":""2019-04-05T17:35:29.809Z 3dd5b57c-4a4a-48aa-8a35-
cbf027b7957e Task timed out after 3.00 seconds""}"
```

```
awsexamplebucket1,image_17644,,failed,200,PermanentFailure,"Lambda
 returned function error: {""errorMessage"":""2019-04-05T17:35:46.025Z
 10a764e4-2b26-4d8c-9056-1e1072b4723f Task timed out after 3.00 seconds""}"
awsexamplebucket1,image_17398,,failed,200,PermanentFailure,"Lambda returned
 function error: {""errorMessage"":""2019-04-05T17:35:44.661Z 1e306352-4c54-4eba-
aee8-4d02f8c0235c Task timed out after 3.00 seconds""}"
```

Example laporan tugas yang berhasil

Laporan tugas yang berhasil berisi hal-hal berikut untuk tugas-tugas yang berhasil:

- `Bucket`
- `Key`
- `VersionId`
- `TaskStatus`
- `ErrorCode`
- `HTTPStatusCode`
- `ResultMessage`

Dalam contoh berikut, fungsi Lambda berhasil menyalin objek Amazon S3 ke bucket lain. Respons Amazon S3 yang dikembalikan akan diteruskan ke Operasi Batch S3 dan kemudian ditulis ke dalam laporan penyelesaian akhir.

```
awsexamplebucket1,image_17775,,succeeded,200,,"{u'CopySourceVersionId':
 'xVR78haVKlRnurYofbTfYr3ufYbktF8h', u'CopyObjectResult': {u'LastModified':
 datetime.datetime(2019, 4, 5, 17, 35, 39, tzinfo=tzlocal()), u'ETag':
 '""fe66f4390c50f29798f040d7aae72784""'}, 'ResponseMetadata': {'HTTPStatusCode':
 200, 'RetryAttempts': 0, 'HostId': 'nXNaClIMxEJzWNmeMNQV2KpjbaCJLn0OGoXWZpuVOFS/
iQYWxb3QtTvzX9SVfx2lA3oTKLwImKw=', 'RequestId': '3ED5852152014362', 'HTTPHeaders':
 {'content-length': '234', 'x-amz-id-2': 'nXNaClIMxEJzWNmeMNQV2KpjbaCJLn0OGoXWZpuVOFS/
iQYWxb3QtTvzX9SVfx2lA3oTKLwImKw=', 'x-amz-copy-source-version-id':
 'xVR78haVKlRnurYofbTfYr3ufYbktF8h', 'server': 'AmazonS3', 'x-amz-request-id':
 '3ED5852152014362', 'date': 'Fri, 05 Apr 2019 17:35:39 GMT', 'content-type':
 'application/xml'}}}"
awsexamplebucket1,image_17763,,succeeded,200,,"{u'CopySourceVersionId':
 '6HjOUSim4Wj6BTcbxToXW44pSZ.40pwq', u'CopyObjectResult': {u'LastModified':
 datetime.datetime(2019, 4, 5, 17, 35, 39, tzinfo=tzlocal()),
 u'ETag': '""fe66f4390c50f29798f040d7aae72784""'}, 'ResponseMetadata':
```

```
 {'HTTPStatusCode': 200, 'RetryAttempts': 0, 'HostId': 'GiCZNYr8LHd/
Thyk6beTRP96IGZk2sYxujLe13TuuLpq6U2RD3we0YoluuIdm1PRvkMwnEW1aFc=', 'RequestId':
 '1BC9F5B1B95D7000', 'HTTPHeaders': {'content-length': '234', 'x-amz-id-2':
 'GiCZNYr8LHd/Thyk6beTRP96IGZk2sYxujLe13TuuLpq6U2RD3we0YoluuIdm1PRvkMwnEW1aFc=', 'x-
amz-copy-source-version-id': '6HjOUSim4Wj6BTcbxToXW44pSZ.40pwq', 'server': 'AmazonS3',
 'x-amz-request-id': '1BC9F5B1B95D7000', 'date': 'Fri, 05 Apr 2019 17:35:39 GMT',
 'content-type': 'application/xml'}}}"
awsexamplebucket1,image_17860,,succeeded,200,,"{u'CopySourceVersionId':
 'm.MDD0g_QsUnYZ8TBzVFrp.TmjN8PJyX', u'CopyObjectResult': {u'LastModified':
 datetime.datetime(2019, 4, 5, 17, 35, 40, tzinfo=tzlocal()), u'ETag':
 '""fe66f4390c50f29798f040d7aae72784""'}, 'ResponseMetadata': {'HTTPStatusCode':
 200, 'RetryAttempts': 0, 'HostId': 'F9ooZOgpE5g9sNgBZxjdiPHqB4+0DNWgj3qbsir
+sKai4fv7rQEcF2fBN1VeeFc2WH45a9ygb2g=', 'RequestId': '8D9CA56A56813DF3', 'HTTPHeaders':
 {'content-length': '234', 'x-amz-id-2': 'F9ooZOgpE5g9sNgBZxjdiPHqB4+0DNWgj3qbsir
+sKai4fv7rQEcF2fBN1VeeFc2WH45a9ygb2g=', 'x-amz-copy-source-version-id':
 'm.MDD0g_QsUnYZ8TBzVFrp.TmjN8PJyX', 'server': 'AmazonS3', 'x-amz-request-id':
 '8D9CA56A56813DF3', 'date': 'Fri, 05 Apr 2019 17:35:40 GMT', 'content-type':
 'application/xml'}}}"
```

Mengendalikan pekerjaan akses dan pelabelan menggunakan tag

Anda dapat melabeli dan mengontrol akses ke pekerjaan S3 Batch Operations Anda dengan menambahkan tag. Tag dapat digunakan untuk mengidentifikasi siapa yang bertanggung jawab atas pekerjaan Batch Operation. Keberadaan tag pekerjaan dapat memberikan atau membatasi kemampuan pengguna untuk membatalkan pekerjaan, mengaktifkan pekerjaan dalam status konfirmasi, atau mengubah tingkat prioritas pekerjaan. Anda dapat membuat pekerjaan dengan tag yang telah dilampirkan, atau menambahkan tag ke pekerjaan setelah pekerjaan dibuat. Setiap tag adalah pasangan nilai kunci yang dapat disertakan saat Anda membuat pekerjaan atau memperbaruinya nanti.

> **Warning**
>
> Tag pekerjaan tidak boleh berisi informasi rahasia atau data pribadi apa pun.

Pertimbangkan contoh tagging berikut: Misalkan Anda ingin departemen Keuangan Anda membuat pekerjaan Batch Operations. Anda bisa menulis (IAM) kebijakanAWS Identity and Access Management yang memungkinkan pengguna untuk memohon `CreateJob`, asalkan tugas dibuat dengan tanda `Department` yang memberikan nilai `Finance`. Selain itu, Anda dapat melampirkan kebijakan tersebut kepada semua pengguna yang merupakan anggota departemen Keuangan.

Melanjutkan dengan contoh ini, Anda dapat menulis kebijakan yang memungkinkan pengguna untuk memperbarui prioritas pekerjaan apa pun yang memiliki tag yang diinginkan, atau membatalkan pekerjaan apa pun yang memiliki tag tersebut. Untuk informasi selengkapnya, lihat the section called "Mengontrol izin".

Anda dapat menambahkan tag ke pekerjaan baru S3 Batch Operations saat Anda membuatnya, atau Anda dapat menambahkannya ke pekerjaan yang sudah ada.

Perhatikan pembatasan tag berikut:

- Anda dapat menghubungkan hingga 50 tag dengan sebuah pekerjaan selama tag tersebut memiliki kunci tag unik.
- Panjang kunci tag dapat terdiri hingga 128 karakter Unicode, dan panjang nilai tag dapat terdiri hingga 256 karakter Unicode.
- Kunci dan nilainya peka tergadap huruf besar dan kecil.

Untuk informasi lebih lanjut tentang pembatasan tanda, lihat Pembatasan Tanda Buatan Pengguna di AWS Billing and Cost Management Panduan Pengguna.

Pengoperasian API yang terkait dengan tagging pekerjaan S3 Batch Operations

Amazon S3 mendukung pengoperasian API berikut yang khusus untuk tagging pekerjaan S3 Batch Operations:

- GetJobTagging — Mengembalikan kumpulan tanda yang terkait dengan tugas Operasi Batch.
- PutJobTagging — Mengganti kumpulan tanda yang terkait dengan tugas. Ada dua skenario berbeda untuk manajemen tag pekerjaan S3 Batch Operations menggunakan tindakan API ini:
  - Pekerjaan tidak memiliki tag — Anda dapat menambahkan rangkaian tag ke pekerjaan (sebelumnya pekerjaan tidak memiliki tag).
  - Pekerjaan memiliki kumpulan tanda yang ada — Untuk mengubah kumpulan tanda yang ada, Anda dapat mengganti kumpulan tanda yang ada seluruhnya, atau membuat perubahan dalam kumpulan tanda yang ada dengan mengambil kumpulan tanda yang ada menggunakan GetJobTagging, memodifikasi kumpulan tanda tersebut, dan menggunakan ini Tindakan API untuk mengganti kumpulan tanda dengan yang telah Anda modifikasi.

> **Note**
>
> Jika Anda mengirim permintaan ini dengan rangkaian tag kosong, S3 Batch Operations akan menghapus rangkaian tag yang sudah ada pada objek. Jika Anda menggunakan metode ini, Anda akan dikenakan biaya untuk Permintaan Tingkat 1 (PUT). Untuk informasi lebih lanjut, lihat harga Amazon S3.
>
> Untuk menghapus tag yang sudah ada di pekerjaan Batch Operations Anda, tindakan `DeleteJobTagging` lebih disukai karena mencapai hasil yang sama tanpa menimbulkan biaya.

- DeleteJobTagging — Menghapus kumpulan tanda yang terkait dengan pekerjaan Operasi Batch.

Membuat tugas Operasi Batch dengan tanda pekerjaan yang digunakan untuk pelabelan

Anda dapat melabeli dan mengontrol akses ke pekerjaan S3 Batch Operations Anda dengan menambahkan tag. Tag dapat digunakan untuk mengidentifikasi siapa yang bertanggung jawab atas pekerjaan Batch Operation. Anda dapat membuat pekerjaan dengan tag yang telah dilampirkan, atau menambahkan tag ke pekerjaan setelah pekerjaan dibuat. Untuk informasi selengkapnya, lihat the section called “Menggunakan tanda”.

Menggunakan AWS CLI

Contoh AWS CLI berikut membuat tugas `S3PutObjectCopy` Operasi Batch S3 menggunakan tanda tugas sebagai label untuk pekerjaan tersebut.

1. Pilih tindakan atau `OPERATION` yang akan dilaksanakan oleh Batch Operations, lalu pilih `TargetResource`.

```
read -d '' OPERATION <<EOF
{
  "S3PutObjectCopy": {
    "TargetResource": "arn:aws:s3:::destination-bucket"
  }
}
EOF
```

2. Identifikasi pekerjaan `TAGS` yang Anda inginkan untuk pekerjaan tersebut. Dalam hal ini, Anda akan menerapkan dua tag, `department` dan `FiscalYear`, masing-masing dengan nilai `Marketing` dan `2020`.

```
read -d '' TAGS <<EOF
[
  {
    "Key": "department",
    "Value": "Marketing"
  },
  {
    "Key": "FiscalYear",
    "Value": "2020"
  }
]
EOF
```

3. Tentukan `MANIFEST` untuk pekerjaan Batch Operations.

```
read -d '' MANIFEST <<EOF
{
  "Spec": {
    "Format": "EXAMPLE_S3BatchOperations_CSV_20180820",
    "Fields": [
      "Bucket",
      "Key"
    ]
  },
  "Location": {
    "ObjectArn": "arn:aws:s3:::example-bucket/example_manifest.csv",
    "ETag": "example-5dc7a8bfb90808fc5d546218"
  }
}
EOF
```

4. Konfigurasikan `REPORT` untuk pekerjaan Batch Operations.

```
read -d '' REPORT <<EOF
{
  "Bucket": "arn:aws:s3:::example-report-bucket",
  "Format": "Example_Report_CSV_20180820",
  "Enabled": true,
  "Prefix": "reports/copy-with-replace-metadata",
```

```
  "ReportScope": "AllTasks"
}
EOF
```

5. Jalankan tindakan `create-job` untuk membuat pekerjaan Batch Operations dengan input yang telah ditetapkan pada langkah sebelumnya.

```
aws \
    s3control create-job \
    --account-id 123456789012 \
    --manifest "${MANIFEST//$'\n'}" \
    --operation "${OPERATION//$'\n'/}" \
    --report "${REPORT//$'\n'}" \
    --priority 10 \
    --role-arn arn:aws:iam::123456789012:role/batch-operations-role \
    --tags "${TAGS//$'\n'/}" \
    --client-request-token "$(uuidgen)" \
    --region us-west-2 \
    --description "Copy with Replace Metadata";
```

MenggunakanAWSSDK for Java

Example

Contoh berikut membuat tugas Operasi Batch S3 dengan tanda menggunakan AWS SDK for Java.

```
public String createJob(final AWSS3ControlClient awss3ControlClient) {
    final String manifestObjectArn = "arn:aws:s3:::example-manifest-bucket/
manifests/10_manifest.csv";
    final String manifestObjectVersionId = "example-5dc7a8bfb90808fc5d546218";

    final JobManifestLocation manifestLocation = new JobManifestLocation()
            .withObjectArn(manifestObjectArn)
            .withETag(manifestObjectVersionId);

    final JobManifestSpec manifestSpec =
            new
 JobManifestSpec().withFormat(JobManifestFormat.S3InventoryReport_CSV_20161130);

    final JobManifest manifestToPublicApi = new JobManifest()
            .withLocation(manifestLocation)
            .withSpec(manifestSpec);
```

```
    final String jobReportBucketArn = "arn:aws:s3:::example-report-bucket";
    final String jobReportPrefix = "example-job-reports";

    final JobReport jobReport = new JobReport()
            .withEnabled(true)
            .withReportScope(JobReportScope.AllTasks)
            .withBucket(jobReportBucketArn)
            .withPrefix(jobReportPrefix)
            .withFormat(JobReportFormat.Report_CSV_20180820);

    final String lambdaFunctionArn = "arn:aws:lambda:us-
west-2:123456789012:function:example-function";

    final JobOperation jobOperation = new JobOperation()
            .withLambdaInvoke(new
 LambdaInvokeOperation().withFunctionArn(lambdaFunctionArn));

    final S3Tag departmentTag = new
 S3Tag().withKey("department").withValue("Marketing");
    final S3Tag fiscalYearTag = new S3Tag().withKey("FiscalYear").withValue("2020");

    final String roleArn = "arn:aws:iam::123456789012:role/example-batch-operations-
role";
    final Boolean requiresConfirmation = true;
    final int priority = 10;

    final CreateJobRequest request = new CreateJobRequest()
            .withAccountId("123456789012")
            .withDescription("Test lambda job")
            .withManifest(manifestToPublicApi)
            .withOperation(jobOperation)
            .withPriority(priority)
            .withRoleArn(roleArn)
            .withReport(jobReport)
            .withTags(departmentTag, fiscalYearTag)
            .withConfirmationRequired(requiresConfirmation);

    final CreateJobResult result = awss3ControlClient.createJob(request);

    return result.getJobId();
}
```

Menghapus tanda dari tugas Operasi Batch S3

Anda dapat menggunakan contoh berikut untuk menghapus tanda dari tugas Operasi Batch.

Menggunakan AWS CLI

Contoh berikut menghapus tanda dari tugas Operasi Batch menggunakan AWS CLI.

```
aws \
    s3control delete-job-tagging \
    --account-id 123456789012 \
    --job-id Example-e25a-4ed2-8bee-7f8ed7fc2f1c \
    --region us-east-1;
```

Menghapus tag pekerjaan dari pekerjaan Batch Operations

Example

Contoh berikut menghapus tanda tugas Operasi Batch S3 menggunakan AWS SDK for Java.

```
public void deleteJobTagging(final AWSS3ControlClient awss3ControlClient,
                             final String jobId) {
    final DeleteJobTaggingRequest deleteJobTaggingRequest = new
 DeleteJobTaggingRequest()
            .withJobId(jobId);

    final DeleteJobTaggingResult deleteJobTaggingResult =
                awss3ControlClient.deleteJobTagging(deleteJobTaggingRequest);
}
```

Menempatkan tanda tugas untuk tugas Operasi Batch S3 yang ada

Anda dapat menggunakan PutJobTagging untuk menambahkan tanda tugas yang sudah ada ke tugas Operasi Batch S3. Untuk informasi selengkapnya, lihat contoh berikut.

Menggunakan AWS CLI

Berikut ini adalah contoh penggunaan `s3control put-job-tagging` untuk menambahkan tanda tugas ke tugas Operasi Batch S3 Anda menggunakan AWS CLI.

> **Note**
>
> Jika Anda mengirim permintaan ini dengan rangkaian tag kosong, S3 Batch Operations akan menghapus rangkaian tag yang sudah ada pada objek. Selain itu, jika Anda menggunakan metode ini, Anda akan dikenakan biaya untuk Permintaan Tingkat 1 (PUT). Untuk informasi lebih lanjut, lihat harga Amazon S3.
>
> Untuk menghapus tag yang sudah ada di pekerjaan Batch Operations Anda, tindakan `DeleteJobTagging` lebih disukai karena mencapai hasil yang sama tanpa menimbulkan biaya.

1. Identifikasi pekerjaan `TAGS` yang Anda inginkan untuk pekerjaan tersebut. Dalam hal ini, Anda akan menerapkan dua tag, `department` dan `FiscalYear`, masing-masing dengan nilai `Marketing` dan `2020`.

```
read -d '' TAGS <<EOF
[
  {
    "Key": "department",
    "Value": "Marketing"
  },
  {
    "Key": "FiscalYear",
    "Value": "2020"
  }
]
EOF
```

2. Jalankan tindakan `put-job-tagging` dengan parameter yang diperlukan.

```
aws \
    s3control put-job-tagging \
    --account-id 123456789012 \
    --tags "${TAGS//$'\n'/}" \
    --job-id Example-e25a-4ed2-8bee-7f8ed7fc2f1c \
    --region us-east-1;
```

MenggunakanAWSSDK for Java

Example

Contoh berikut menempatkan tanda tugas Operasi Batch S3 menggunakan AWS SDK for Java.

```
public void putJobTagging(final AWSS3ControlClient awss3ControlClient,
                          final String jobId) {
    final S3Tag departmentTag = new
 S3Tag().withKey("department").withValue("Marketing");
    final S3Tag fiscalYearTag = new S3Tag().withKey("FiscalYear").withValue("2020");

    final PutJobTaggingRequest putJobTaggingRequest = new PutJobTaggingRequest()
            .withJobId(jobId)
            .withTags(departmentTag, fiscalYearTag);

    final PutJobTaggingResult putJobTaggingResult =
 awss3ControlClient.putJobTagging(putJobTaggingRequest);
}
```

Mendapatkan tanda dari tugas Operasi Batch S3

Anda dapat menggunakan `GetJobTagging` untuk mengembalikan tanda tugas Operasi Batch S3. Untuk informasi selengkapnya, lihat contoh berikut.

Menggunakan AWS CLI

Contoh berikut menunjukkan cara mendapatkan tanda tugas Operasi Batch menggunakan AWS CLI.

```
aws \
    s3control get-job-tagging \
    --account-id 123456789012 \
    --job-id Example-e25a-4ed2-8bee-7f8ed7fc2f1c \
    --region us-east-1;
```

MenggunakanAWSSDK for Java

Example

Contoh berikut menunjukkan cara mendapatkan tanda tugas Operasi Batch S3 menggunakan AWS SDK for Java.

```
public List<S3Tag> getJobTagging(final AWSS3ControlClient awss3ControlClient,
```

```
                                final String jobId) {
    final GetJobTaggingRequest getJobTaggingRequest = new GetJobTaggingRequest()
            .withJobId(jobId);

    final GetJobTaggingResult getJobTaggingResult =
            awss3ControlClient.getJobTagging(getJobTaggingRequest);

    final List<S3Tag> tags = getJobTaggingResult.getTags();

    return tags;
}
```

Mengontrol izin untuk Operasi Batch S3 menggunakan tanda tugas

Untuk membantu mengelola tugas Operasi Batch S3, Anda dapat menambahkan tanda tugas. Dengan tanda tugas, Anda dapat mengontrol akses atas tugas Operasi Batch dan menerapkan tanda tersebut saat pekerjaan lain dibuat.

Anda dapat menerapkan hingga 50 tanda tugas untuk setiap tugas Operasi Batch. Dengan cara ini, Anda dapat menetapkan kebijakan yang sangat terperinci yang membatasi kumpulan pengguna yang dapat mengedit pekerjaan. Tanda tugas dapat memberikan atau membatasi kemampuan pengguna untuk membatalkan tugas, mengaktifkan tugas dalam status konfirmasi, atau mengubah tingkat prioritas tugas. Selain itu, Anda dapat memberlakukan tanda yang diterapkan ke semua tugas baru, dan menentukan pasangan nilai kunci yang diizinkan untuk tanda tersebut. Anda dapat menerapkan semua ketentuan ini menggunakan bahasa kebijakan IAM yang sama. Untuk informasi selengkapnya, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon S3 di Referensi Otorisasi Layanan.

Contoh berikut menunjukkan cara menggunakan tanda tugas Operasi Batch S3 untuk mengizinkan pengguna dalam membuat dan mengedit tugas yang khusus dijalankan di dalam departemen (misalnya, departemen Keuangan atau Kepatuhan). Anda juga dapat menentukan tugas berdasarkan tahap pengembangan yang terkait dengan tugas tersebut, seperti QA atau Produksi.

Dalam contoh ini, Anda menggunakan tag pekerjaan Operasi Batch S3 dalam kebijakan AWS Identity and Access Management (IAM) untuk memberikan izin kepada pengguna untuk membuat dan mengedit hanya pekerjaan yang dijalankan di departemen mereka. Tentukan pekerjaan berdasarkan tahap pengembangan yang terkait dengan pekerjaan tersebut, seperti QA atau Produksi.

Contoh ini menggunakan departemen berikut, yang masing-masing menggunakan Operasi Batch dengan cara berbeda:

- Keuangan
- Kepatuhan
- Kecerdasan Bisnis
- Teknik

Topik

- Mengontrol akses dengan menetapkan tanda ke pengguna dan sumber daya
- Menandai pekerjaan Operasi Batch berdasarkan tahap dan memberlakukan batas prioritas pekerjaan

Mengontrol akses dengan menetapkan tanda ke pengguna dan sumber daya

Dalam skenario ini, administrator menggunakan kontrol akses berbasis atribut (ABAC). ABAC adalah strategi otorisasi IAM yang mendefinisikan izin dengan melampirkan tag ke pengguna dan sumber daya. AWS

Pengguna dan tugas diberi salah satu tanda departemen berikut ini:

Kunci : Nilai

- `department : Finance`
- `department : Compliance`
- `department : BusinessIntelligence`
- `department : Engineering`

> **Note**
>
> Kunci dan nilai tanda tugas peka huruf besar kecil.

Dengan menggunakan strategi kontrol akses ABAC, Anda mengizinkan pengguna di departemen Keuangan untuk membuat dan mengelola pekerjaan Operasi Batch S3 dalam departemen mereka dengan mengasosiasikan tanda `department=Finance` dengan pengguna mereka.

Selain itu, Anda dapat melampirkan kebijakan terkelola ke pengguna IAM yang memungkinkan setiap pengguna di perusahaan untuk membuat atau memodifikasi tugas Operasi Batch S3 dalam departemen mereka masing-masing.

Kebijakan dalam contoh ini mencakup tiga pernyataan kebijakan:

- Pernyataan pertama dalam kebijakan mengizinkan pengguna untuk membuat tugas Operasi Batch asalkan permintaan pembuatan tugas menyertakan tanda tugas yang sesuai dengan departemen masing-masing. Ini ditunjukkan dengan menggunakan sintaks `"${aws:PrincipalTag/department}"`, yang diganti dengan tanda departemen pengguna pada waktu evaluasi kebijakan. Syarat ini terpenuhi ketika nilai yang diberikan untuk tanda departemen dalam permintaan (`"aws:RequestTag/department"`) sesuai dengan departemen pengguna.
- Pernyataan kedua dalam kebijakan ini memungkinkan pengguna untuk mengubah prioritas tugas atau memperbarui statusnya asalkan tugas yang sedang diperbarui sesuai dengan departemen pengguna.
- Pernyataan ketiga memungkinkan pengguna untuk memperbarui tanda tugas Operasi Batch setiap saat melalui permintaan `PutJobTagging` asalkan (1) tanda departemen mereka dipertahankan dan (2) pekerjaan yang diperbarui berada dalam departemen mereka.

```
{
      "Version": "2012-10-17",
      "Statement": [
            {
                  "Effect": "Allow",
                  "Action": "s3:CreateJob",
                  "Resource": "*",
                  "Condition": {
                        "StringEquals": {
                              "aws:RequestTag/department": "${aws:PrincipalTag/
department}"
                }
            }
        },
            {
                  "Effect": "Allow",
                  "Action": [
                        "s3:UpdateJobPriority",
                        "s3:UpdateJobStatus"
            ],
                  "Resource": "*",
                  "Condition": {
                        "StringEquals": {
                              "aws:ResourceTag/department": "${aws:PrincipalTag/
department}"
```

```
                }
            }
        },
            {
                "Effect": "Allow",
                "Action": "s3:PutJobTagging",
                "Resource": "*",
                "Condition": {
                    "StringEquals": {
                        "aws:RequestTag/department": "${aws:PrincipalTag/
department}",
                        "aws:ResourceTag/department": "${aws:PrincipalTag/
department}"
                }
            }
        }
    ]
}
```

Menandai pekerjaan Operasi Batch berdasarkan tahap dan memberlakukan batas prioritas pekerjaan

Semua pekerjaan Operasi Batch S3 memiliki prioritas numerik, yang digunakan Amazon S3 untuk memutuskan urutan pelaksanaan tugas. Untuk contoh ini, Anda membatasi prioritas maksimum yang dapat ditetapkan oleh sebagian besar pengguna ke tugas, dengan rentang prioritas lebih tinggi yang diperuntukkan bagi pengguna dengan hak istimewa dalam jumlah terbatas, sebagai berikut:

- Kisaran prioritas tahap QA (rendah): 1-100
- Kisaran prioritas tahap produksi (tinggi): 1-300

Untuk melakukannya, gunakan set tanda baru yang mewakili tahapan tugas:

Kunci : Nilai

- `stage : QA`
- `stage : Production`

Membuat dan memperbarui pekerjaan prioritas rendah dalam sebuah departemen

Kebijakan ini memperkenalkan dua pembatasan baru pada pembuatan dan pembaruan tugas Operasi Batch S3, selain pembatasan berbasis departemen:

- Ini memungkinkan pengguna membuat atau memperbarui tugas di departemen mereka dengan ketentuan baru yang memerlukan tugas tersebut untuk menyertakan tanda `stage=QA`.
- Ini memungkinkan pengguna membuat atau memperbarui prioritas tugas hingga prioritas maksimum baru sebesar 100.

```
{
        "Version": "2012-10-17",
        "Statement": [
        {
        "Effect": "Allow",
        "Action": "s3:CreateJob",
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/department": "${aws:PrincipalTag/department}",
                "aws:RequestTag/stage": "QA"
            },
            "NumericLessThanEquals": {
                "s3:RequestJobPriority": 100
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "s3:UpdateJobStatus"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/department": "${aws:PrincipalTag/department}"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": "s3:UpdateJobPriority",
        "Resource": "*",
```

```
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/department": "${aws:PrincipalTag/department}",
                "aws:ResourceTag/stage": "QA"
            },
            "NumericLessThanEquals": {
                "s3:RequestJobPriority": 100
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": "s3:PutJobTagging",
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/department" : "${aws:PrincipalTag/department}",
                "aws:ResourceTag/department": "${aws:PrincipalTag/department}",
                "aws:RequestTag/stage": "QA",
                "aws:ResourceTag/stage": "QA"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": "s3:GetJobTagging",
        "Resource": "*"
    }
    ]
}
```

Membuat dan memperbarui tugas prioritas tinggi dalam sebuah departemen

Sebagian kecil pengguna mungkin memerlukan kemampuan untuk membuat tugas dengan prioritas tinggi di QA maupun Produksi. Untuk mendukung kebutuhan ini, buat kebijakan terkelola yang diadaptasi dari kebijakan prioritas rendah pada bagian sebelumnya.

Kebijakan ini mengatur hal-hal berikut:

- Memungkinkan pengguna membuat atau memperbarui tugas di departemen mereka dengan tanda `stage=QA` atau `stage=Production`.
- Memungkinkan pengguna membuat atau memperbarui prioritas tigas hingga maksimum 300.

```
{
      "Version": "2012-10-17",
      "Statement": [
          {
                "Effect": "Allow",
                "Action": "s3:CreateJob",
                "Resource": "*",
                "Condition": {
                      "ForAnyValue:StringEquals": {
                            "aws:RequestTag/stage": [
                                  "QA",
                                  "Production"
                    ]
                },
                      "StringEquals": {
                            "aws:RequestTag/department": "${aws:PrincipalTag/
department}"
                },
                      "NumericLessThanEquals": {
                            "s3:RequestJobPriority": 300
                }
            }
        },
          {
                "Effect": "Allow",
                "Action": [
                      "s3:UpdateJobStatus"
            ],
                "Resource": "*",
                "Condition": {
                      "StringEquals": {
                            "aws:ResourceTag/department": "${aws:PrincipalTag/
department}"
                }
            }
        },
          {
                "Effect": "Allow",
                "Action": "s3:UpdateJobPriority",
                "Resource": "*",
                "Condition": {
                      "ForAnyValue:StringEquals": {
                            "aws:ResourceTag/stage": [
```

```
                                "QA",
                                "Production"
                    ]
                },
                      "StringEquals": {
                            "aws:ResourceTag/department": "${aws:PrincipalTag/
department}"
                },
                      "NumericLessThanEquals": {
                            "s3:RequestJobPriority": 300
                }
            }
        },
          {
                "Effect": "Allow",
                "Action": "s3:PutJobTagging",
                "Resource": "*",
                "Condition": {
                      "StringEquals": {
                            "aws:RequestTag/department": "${aws:PrincipalTag/
department}",
                            "aws:ResourceTag/department": "${aws:PrincipalTag/
department}"
                },
                      "ForAnyValue:StringEquals": {
                            "aws:RequestTag/stage": [
                                  "QA",
                                  "Production"
                    ],
                            "aws:ResourceTag/stage": [
                                  "QA",
                                  "Production"
                    ]
                }
            }
        }
    ]
}
```

Mengelola Kunci Objek S3 menggunakan Operasi Batch S3

Dengan Kunci Objek S3, Anda dapat menambahkan penahanan hukum pada versi objek. Seperti mengatur periode retensi, penyimpanan yang sah mencegah agar versi objek tidak ditimpa atau

dihapus. Namun, penyimpanan yang sah tidak memiliki periode retensi terkait dan akan tetap berlaku hingga dihapus. Untuk informasi selengkapnya, lihat Pegangan hukum S3 Object Lock.

Untuk informasi tentang menggunakan Operasi Batch S3 dengan Kunci Objek untuk menambahkan penahanan hukum ke banyak objek Amazon S3 sekaligus, lihat bagian berikut.

Topik

- Mengaktifkan Kunci Objek S3 menggunakan Operasi Batch S3
- Mengatur retensi Kunci Objek menggunakan Operasi Batch
- Menggunakan Operasi Batch S3 dengan mode kepatuhan retensi Kunci Objek S3
- Gunakan Operasi Batch S3 dengan mode tata kelola penyimpanan Kunci Objek S3
- Menggunakan Operasi Batch S3 untuk menonaktifkan penahanan hukum Kunci Objek S3

Mengaktifkan Kunci Objek S3 menggunakan Operasi Batch S3

Anda dapat menggunakan Operasi Batch S3 dengan Kunci Objek S3 untuk mengelola penyimpanan atau mengaktifkan penahanan hukum untuk banyak objek Amazon S3 sekaligus. Anda menetapkan daftar objek target dalam manifes dan mengirimkannya ke Operasi Batch untuk penyelesaian. Untuk informasi lebih lanjut, lihat the section called “Retensi Object Lock” dan the section called “Pegangan hukum Object Lock”.

Contoh berikut menunjukkan cara membuat peran IAM dengan izin Operasi Batch S3 dan memperbarui izin peran untuk membuat pekerjaan yang akan mengaktifkan Kunci Objek. Dalam contoh, ganti nilai variabel dengan nilai yang sesuai dengan kebutuhan Anda. Anda juga harus memiliki manifes `CSV` yang mengidentifikasi objek untuk pekerjaan Operasi Batch S3 Anda. Untuk informasi selengkapnya, lihat the section called “Menentukan manifes”.

Menggunakan AWS CLI

1. Membuat peran IAM dan menetapkan izin Operasi Batch S3 yang akan dijalankan.

   Langkah ini diperlukan untuk semua pekerjaan Operasi Batch S3.

```
export AWS_PROFILE='aws-user'

read -d '' bops_trust_policy <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "batchoperations.s3.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
aws iam create-role --role-name bops-objectlock --assume-role-policy-document
 "${bops_trust_policy}"
```

2. Menetapkan Operasi Batch S3 dengan Kunci Objek S3 yang akan dijalankan.

   Pada langkah ini, izinkan peran untuk melakukan hal-hal berikut ini:

   a. Menjalankan Kunci Objek pada bucket S3 yang berisi objek target yang akan dijalankan oleh Operasi Batch.

   b. Membaca bucket S3 tempat file CSV manifes dan objeknya berada.

   c. Menulis hasil pekerjaan Operasi Batch S3 ke bucket pelaporan.

```
read -d '' bops_permissions <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetBucketObjectLockConfiguration",
            "Resource": [
                "arn:aws:s3:::{{ManifestBucket}}"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:GetBucketLocation"
            ],
```

```
            "Resource": [
                "arn:aws:s3:::{{ManifestBucket}}/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::{{ReportBucket}}/*"
            ]
        }
    ]
}
EOF

aws iam put-role-policy --role-name bops-objectlock --policy-name object-lock-
permissions --policy-document "${bops_permissions}"
```

Menggunakan AWS SDK for Java

Contoh berikut menunjukkan cara membuat peran IAM dengan izin Operasi Batch S3, dan memperbarui izin peran untuk membuat pekerjaan yang mengaktifkan kunci objek menggunakan AWS SDK for Java. Di dalam kode tersebut, ganti nilai variabel dengan nilai yang sesuai dengan kebutuhan Anda. Anda juga harus memiliki manifes `CSV` yang mengidentifikasi objek untuk pekerjaan Operasi Batch S3 Anda. Untuk informasi selengkapnya, lihat the section called “Menentukan manifes”.

Anda harus melakukan langkah-langkah berikut ini:

1. Membuat peran IAM dan menetapkan izin Operasi Batch S3 yang akan dijalankan. Langkah ini diperlukan untuk semua pekerjaan Operasi Batch S3.
2. Menetapkan Operasi Batch S3 dengan Kunci Objek S3 yang akan dijalankan.

   Izinkan peran tersebut untuk melakukan hal-hal berikut ini:

   1. Menjalankan Kunci Objek pada bucket S3 yang berisi objek target yang akan dijalankan oleh Operasi Batch.
   2. Membaca bucket S3 tempat file CSV manifes dan objeknya berada.

3. Menulis hasil pekerjaan Operasi Batch S3 ke bucket pelaporan.

```
public void createObjectLockRole() {
    final String roleName = "bops-object-lock";

    final String trustPolicy = "{" +
            "  \"Version\": \"2012-10-17\", " +
            "  \"Statement\": [ " +
            "    { " +
            "      \"Effect\": \"Allow\", " +
            "      \"Principal\": { " +
            "        \"Service\": [" +
            "          \"batchoperations.s3.amazonaws.com\"" +
            "        ]" +
            "      }, " +
            "      \"Action\": \"sts:AssumeRole\" " +
            "    } " +
            "  ]" +
            "}";

    final String bopsPermissions = "{" +
            "    \"Version\": \"2012-10-17\"," +
            "    \"Statement\": [" +
            "        {" +
            "            \"Effect\": \"Allow\"," +
            "            \"Action\": \"s3:GetBucketObjectLockConfiguration\"," +
            "            \"Resource\": [" +
            "                \"arn:aws:s3:::ManifestBucket\"" +
            "            ]" +
            "        }," +
            "        {" +
            "            \"Effect\": \"Allow\"," +
            "            \"Action\": [" +
            "                \"s3:GetObject\"," +
            "                \"s3:GetObjectVersion\"," +
            "                \"s3:GetBucketLocation\"" +
            "            ]," +
            "            \"Resource\": [" +
            "                \"arn:aws:s3:::ManifestBucket/*\"" +
            "            ]" +
            "        }," +
            "        {" +
```

```
            "            \"Effect\": \"Allow\"," +
            "            \"Action\": [" +
            "                \"s3:PutObject\"," +
            "                \"s3:GetBucketLocation\"" +
            "            ]," +
            "            \"Resource\": [" +
            "                \"arn:aws:s3:::ReportBucket/*\"" +
            "            ]" +
            "        }" +
            "    ]" +
            "}";

    final AmazonIdentityManagement iam =
            AmazonIdentityManagementClientBuilder.defaultClient();

    final CreateRoleRequest createRoleRequest = new CreateRoleRequest()
            .withAssumeRolePolicyDocument(bopsPermissions)
            .withRoleName(roleName);

    final CreateRoleResult createRoleResult = iam.createRole(createRoleRequest);

    final PutRolePolicyRequest putRolePolicyRequest = new PutRolePolicyRequest()
            .withPolicyDocument(bopsPermissions)
            .withPolicyName("bops-permissions")
            .withRoleName(roleName);

    final PutRolePolicyResult putRolePolicyResult =
 iam.putRolePolicy(putRolePolicyRequest);
}
```

Mengatur retensi Kunci Objek menggunakan Operasi Batch

Contoh berikut ini memungkinkan aturan untuk mengatur retensi Kunci Objek S3 untuk objek Anda di bucket manifes.

Perbarui peran untuk memasukkan izin `s3:PutObjectRetention` sehingga Anda dapat menjalankan retensi Kunci Objek pada objek di bucket Anda.

Menggunakan AWS CLI

```
export AWS_PROFILE='aws-user'

read -d '' retention_permissions <<EOF
```

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObjectRetention"
            ],
            "Resource": [
                "arn:aws:s3:::{{ManifestBucket}}/*"
            ]
        }
    ]
}
EOF

aws iam put-role-policy --role-name bops-objectlock --policy-name retention-permissions
 --policy-document "${retention_permissions}"
```

Menggunakan AWS SDK for Java

```
public void allowPutObjectRetention() {
    final String roleName = "bops-object-lock";

    final String retentionPermissions = "{" +
            "    \"Version\": \"2012-10-17\"," +
            "    \"Statement\": [" +
            "        {" +
            "            \"Effect\": \"Allow\"," +
            "            \"Action\": [" +
            "                \"s3:PutObjectRetention\"" +
            "            ]," +
            "            \"Resource\": [" +
            "                \"arn:aws:s3:::ManifestBucket*\"" +
            "            ]" +
            "        }" +
            "    ]" +
            "}";

    final AmazonIdentityManagement iam =
            AmazonIdentityManagementClientBuilder.defaultClient();

    final PutRolePolicyRequest putRolePolicyRequest = new PutRolePolicyRequest()
```

```
            .withPolicyDocument(retentionPermissions)
            .withPolicyName("retention-permissions")
            .withRoleName(roleName);

    final PutRolePolicyResult putRolePolicyResult =
 iam.putRolePolicy(putRolePolicyRequest);
}
```

Menggunakan Operasi Batch S3 dengan mode kepatuhan retensi Kunci Objek S3

Contoh berikut dibangun berdasarkan contoh sebelumnya tentang pembuatan kebijakan kepercayaan dan pengaturan izin konfigurasi Operasi Batch S3 dan Kunci Objek S3 pada objek Anda. Contoh ini menetapkan mode retensi ke `COMPLIANCE` dan `retain until date` ke 1 Januari 2025. Contoh ini membuat pekerjaan yang menargetkan objek dalam manifes bucket dan melaporkan hasilnya di bucket laporan yang Anda identifikasi.

Menggunakan AWS CLI

Example Tetapkan kepatuhan penyebutan pada beberapa objek

```
export AWS_PROFILE='aws-user'
export AWS_DEFAULT_REGION='us-west-2'
export ACCOUNT_ID=123456789012
export ROLE_ARN='arn:aws:iam::123456789012:role/bops-objectlock'

read -d '' OPERATION <<EOF
{
  "S3PutObjectRetention": {
    "Retention": {
      "RetainUntilDate":"2025-01-01T00:00:00",
      "Mode":"COMPLIANCE"
    }
  }
}
EOF

read -d '' MANIFEST <<EOF
{
  "Spec": {
    "Format": "S3BatchOperations_CSV_20180820",
    "Fields": [
      "Bucket",
      "Key"
```

```
    ]
  },
  "Location": {
    "ObjectArn": "arn:aws:s3:::ManifestBucket/compliance-objects-manifest.csv",
    "ETag": "Your-manifest-ETag"
  }
}
EOF

read -d '' REPORT <<EOF
{
  "Bucket": "arn:aws:s3:::ReportBucket",
  "Format": "Report_CSV_20180820",
  "Enabled": true,
  "Prefix": "reports/compliance-objects-bops",
  "ReportScope": "AllTasks"
}
EOF

aws \
    s3control create-job \
    --account-id "${ACCOUNT_ID}" \
    --manifest "${MANIFEST//$'\n'}" \
    --operation "${OPERATION//$'\n'/}" \
    --report "${REPORT//$'\n'}" \
    --priority 10 \
    --role-arn "${ROLE_ARN}" \
    --client-request-token "$(uuidgen)" \
    --region "${AWS_DEFAULT_REGION}" \
    --description "Set compliance retain-until to 1 Jul 2030";
```

Example Perpanjang **`COMPLIANCE`** mode **`retain until date`** hingga 15 Januari 2025

Contoh berikut memperpanjang `retain until date` mode `COMPLIANCE` hingga 15 Januari 2025.

```
export AWS_PROFILE='aws-user'
export AWS_DEFAULT_REGION='us-west-2'
export ACCOUNT_ID=123456789012
export ROLE_ARN='arn:aws:iam::123456789012:role/bops-objectlock'

read -d '' OPERATION <<EOF
{
  "S3PutObjectRetention": {
    "Retention": {
```

```
      "RetainUntilDate":"2025-01-15T00:00:00",
      "Mode":"COMPLIANCE"
    }
  }
}
EOF

read -d '' MANIFEST <<EOF
{
  "Spec": {
    "Format": "S3BatchOperations_CSV_20180820",
    "Fields": [
      "Bucket",
      "Key"
    ]
  },
  "Location": {
    "ObjectArn": "arn:aws:s3:::ManifestBucket/compliance-objects-manifest.csv",
    "ETag": "Your-manifest-ETag"
  }
}
EOF

read -d '' REPORT <<EOF
{
  "Bucket": "arn:aws:s3:::ReportBucket",
  "Format": "Report_CSV_20180820",
  "Enabled": true,
  "Prefix": "reports/compliance-objects-bops",
  "ReportScope": "AllTasks"
}
EOF

aws \
    s3control create-job \
    --account-id "${ACCOUNT_ID}" \
    --manifest "${MANIFEST//$'\n'}" \
    --operation "${OPERATION//$'\n'/}" \
    --report "${REPORT//$'\n'}" \
    --priority 10 \
    --role-arn "${ROLE_ARN}" \
    --client-request-token "$(uuidgen)" \
    --region "${AWS_DEFAULT_REGION}" \
```

```
    --description "Extend compliance retention to 15 Jan 2025";
```

Menggunakan AWS SDK for Java

Example Atur mode retensi ke COMPLIANCE dan simpan hingga tanggal 1 Januari 2025.

```
public String createComplianceRetentionJob(final AWSS3ControlClient awss3ControlClient)
 throws ParseException {
    final String manifestObjectArn = "arn:aws:s3:::ManifestBucket/compliance-objects-
manifest.csv";
    final String manifestObjectVersionId = "your-object-version-Id";

    final JobManifestLocation manifestLocation = new JobManifestLocation()
            .withObjectArn(manifestObjectArn)
            .withETag(manifestObjectVersionId);

    final JobManifestSpec manifestSpec =
            new JobManifestSpec()
                    .withFormat(JobManifestFormat.S3BatchOperations_CSV_20180820)
                    .withFields("Bucket", "Key");

    final JobManifest manifestToPublicApi = new JobManifest()
            .withLocation(manifestLocation)
            .withSpec(manifestSpec);

    final String jobReportBucketArn = "arn:aws:s3:::ReportBucket";
    final String jobReportPrefix = "reports/compliance-objects-bops";

    final JobReport jobReport = new JobReport()
            .withEnabled(true)
            .withReportScope(JobReportScope.AllTasks)
            .withBucket(jobReportBucketArn)
            .withPrefix(jobReportPrefix)
            .withFormat(JobReportFormat.Report_CSV_20180820);

    final SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");
    final Date janFirst = format.parse("01/01/2025");

    final JobOperation jobOperation = new JobOperation()
            .withS3PutObjectRetention(new S3SetObjectRetentionOperation()
                    .withRetention(new S3Retention()
                            .withMode(S3ObjectLockRetentionMode.COMPLIANCE)
                            .withRetainUntilDate(janFirst)));
```

```
    final String roleArn = "arn:aws:iam::123456789012:role/bops-object-lock";
    final Boolean requiresConfirmation = true;
    final int priority = 10;

    final CreateJobRequest request = new CreateJobRequest()
            .withAccountId("123456789012")
            .withDescription("Set compliance retain-until to 1 Jan 2025")
            .withManifest(manifestToPublicApi)
            .withOperation(jobOperation)
            .withPriority(priority)
            .withRoleArn(roleArn)
            .withReport(jobReport)
            .withConfirmationRequired(requiresConfirmation);

    final CreateJobResult result = awss3ControlClient.createJob(request);

    return result.getJobId();
}
```

Example Memperpanjang **`retain until date`** mode **`COMPLIANCE`**

Contoh berikut memperpanjang `retain until date` mode `COMPLIANCE` hingga 15 Januari 2025.

```
public String createExtendComplianceRetentionJob(final AWSS3ControlClient
 awss3ControlClient) throws ParseException {
    final String manifestObjectArn = "arn:aws:s3:::ManifestBucket/compliance-objects-
manifest.csv";
    final String manifestObjectVersionId = "15ad5ba069e6bbc465c77bf83d541385";

    final JobManifestLocation manifestLocation = new JobManifestLocation()
            .withObjectArn(manifestObjectArn)
            .withETag(manifestObjectVersionId);

    final JobManifestSpec manifestSpec =
            new JobManifestSpec()
                    .withFormat(JobManifestFormat.S3BatchOperations_CSV_20180820)
                    .withFields("Bucket", "Key");

    final JobManifest manifestToPublicApi = new JobManifest()
            .withLocation(manifestLocation)
            .withSpec(manifestSpec);

    final String jobReportBucketArn = "arn:aws:s3:::ReportBucket";
    final String jobReportPrefix = "reports/compliance-objects-bops";
```

```
    final JobReport jobReport = new JobReport()
            .withEnabled(true)
            .withReportScope(JobReportScope.AllTasks)
            .withBucket(jobReportBucketArn)
            .withPrefix(jobReportPrefix)
            .withFormat(JobReportFormat.Report_CSV_20180820);

    final SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");
    final Date jan15th = format.parse("15/01/2025");

    final JobOperation jobOperation = new JobOperation()
            .withS3PutObjectRetention(new S3SetObjectRetentionOperation()
                    .withRetention(new S3Retention()
                            .withMode(S3ObjectLockRetentionMode.COMPLIANCE)
                            .withRetainUntilDate(jan15th)));

    final String roleArn = "arn:aws:iam::123456789012:role/bops-object-lock";
    final Boolean requiresConfirmation = true;
    final int priority = 10;

    final CreateJobRequest request = new CreateJobRequest()
            .withAccountId("123456789012")
            .withDescription("Extend compliance retention to 15 Jan 2025")
            .withManifest(manifestToPublicApi)
            .withOperation(jobOperation)
            .withPriority(priority)
            .withRoleArn(roleArn)
            .withReport(jobReport)
            .withConfirmationRequired(requiresConfirmation);

    final CreateJobResult result = awss3ControlClient.createJob(request);

    return result.getJobId();
}
```

Gunakan Operasi Batch S3 dengan mode tata kelola penyimpanan Kunci Objek S3

Contoh berikut dibangun berdasarkan contoh sebelumnya tentang pembuatan kebijakan kepercayaan, dan pengaturan izin konfigurasi Operasi Batch S3 serta Kunci Objek S3. Contoh ini menunjukkan cara menerapkan tata kelola retensi Kunci Objek S3 dengan `retain until date` hingga 30 Januari 2025, pada beberapa objek. Contoh ini membuat pekerjaan Operasi Batch yang menggunakan bucket manifes dan melaporkan hasilnya di bucket laporan.

Menggunakan AWS CLI

Example Terapkan tata kelola retensi Kunci Objek S3 di beberapa objek dengan penyimpanan hingga tanggal 30 Januari 2025

```
export AWS_PROFILE='aws-user'
export AWS_DEFAULT_REGION='us-west-2'
export ACCOUNT_ID=123456789012
export ROLE_ARN='arn:aws:iam::123456789012:role/bops-objectlock'

read -d '' OPERATION <<EOF
{
  "S3PutObjectRetention": {
    "Retention": {
      "RetainUntilDate":"2025-01-30T00:00:00",
      "Mode":"GOVERNANCE"
    }
  }
}
EOF

read -d '' MANIFEST <<EOF
{
  "Spec": {
    "Format": "S3BatchOperations_CSV_20180820",
    "Fields": [
      "Bucket",
      "Key"
    ]
  },
  "Location": {
    "ObjectArn": "arn:aws:s3:::ManifestBucket/governance-objects-manifest.csv",
    "ETag": "Your-manifest-ETag"
  }
}
EOF

read -d '' REPORT <<EOF
{
  "Bucket": "arn:aws:s3:::ReportBucketT",
  "Format": "Report_CSV_20180820",
  "Enabled": true,
  "Prefix": "reports/governance-objects",
  "ReportScope": "AllTasks"
```

```
}
EOF

aws \
    s3control create-job \
    --account-id "${ACCOUNT_ID}" \
    --manifest "${MANIFEST//$'\n'}" \
    --operation "${OPERATION//$'\n'/}" \
    --report "${REPORT//$'\n'}" \
    --priority 10 \
    --role-arn "${ROLE_ARN}" \
    --client-request-token "$(uuidgen)" \
    --region "${AWS_DEFAULT_REGION}" \
    --description "Put governance retention";
```

Example Memintas tata kelola retensi pada beberapa objek

Contoh berikut dibangun berdasarkan contoh sebelumnya tentang pembuatan kebijakan kepercayaan, dan pengaturan izin konfigurasi Operasi Batch S3 serta Kunci Objek S3. Contoh ini menunjukkan cara memintas tata kelola retensi pada beberapa objek dan membuat pekerjaan Operasi Batch yang menggunakan bucket manifes dan melaporkan hasil di bucket laporan.

```
export AWS_PROFILE='aws-user'

read -d '' bypass_governance_permissions <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:BypassGovernanceRetention"
            ],
            "Resource": [
                "arn:aws:s3:::ManifestBucket/*"
            ]
        }
    ]
}
EOF

aws iam put-role-policy --role-name bops-objectlock --policy-name bypass-governance-
permissions --policy-document "${bypass_governance_permissions}"
```

```
export AWS_PROFILE='aws-user'
export AWS_DEFAULT_REGION='us-west-2'
export ACCOUNT_ID=123456789012
export ROLE_ARN='arn:aws:iam::123456789012:role/bops-objectlock'

read -d '' OPERATION <<EOF
{
  "S3PutObjectRetention": {
    "BypassGovernanceRetention": true,
    "Retention": {
    }
  }
}
EOF

read -d '' MANIFEST <<EOF
{
  "Spec": {
    "Format": "S3BatchOperations_CSV_20180820",
    "Fields": [
      "Bucket",
      "Key"
    ]
  },
  "Location": {
    "ObjectArn": "arn:aws:s3:::ManifestBucket/governance-objects-manifest.csv",
    "ETag": "Your-manifest-ETag"
  }
}
EOF

read -d '' REPORT <<EOF
{
  "Bucket": "arn:aws:s3:::REPORT_BUCKET",
  "Format": "Report_CSV_20180820",
  "Enabled": true,
  "Prefix": "reports/bops-governance",
  "ReportScope": "AllTasks"
}
EOF

aws \
    s3control create-job \
```

```
    --account-id "${ACCOUNT_ID}" \
    --manifest "${MANIFEST//$'\n'}" \
    --operation "${OPERATION//$'\n'/}" \
    --report "${REPORT//$'\n'}" \
    --priority 10 \
    --role-arn "${ROLE_ARN}" \
    --client-request-token "$(uuidgen)" \
    --region "${AWS_DEFAULT_REGION}" \
    --description "Remove governance retention";
```

Menggunakan AWS SDK for Java

Contoh berikut dibangun berdasarkan contoh sebelumnya tentang pembuatan kebijakan kepercayaan, dan pengaturan izin konfigurasi Operasi Batch S3 serta Kunci Objek S3. Ini menunjukkan cara menerapkan tata kelola retensi Kunci Objek S3 dengan `retain until date` set ke 30 Januari 2025 di beberapa objek. Contoh ini membuat pekerjaan Operasi Batch yang menggunakan bucket manifes dan melaporkan hasilnya di bucket laporan.

Example Terapkan tata kelola retensi Kunci Objek S3 di beberapa objek dengan penyimpanan hingga tanggal 30 Januari 2025

```
public String createGovernanceRetentionJob(final AWSS3ControlClient awss3ControlClient)
 throws ParseException {
    final String manifestObjectArn = "arn:aws:s3:::ManifestBucket/governance-objects-
manifest.csv";
    final String manifestObjectVersionId = "15ad5ba069e6bbc465c77bf83d541385";

    final JobManifestLocation manifestLocation = new JobManifestLocation()
            .withObjectArn(manifestObjectArn)
            .withETag(manifestObjectVersionId);

    final JobManifestSpec manifestSpec =
            new JobManifestSpec()
                    .withFormat(JobManifestFormat.S3BatchOperations_CSV_20180820)
                    .withFields("Bucket", "Key");

    final JobManifest manifestToPublicApi = new JobManifest()
            .withLocation(manifestLocation)
            .withSpec(manifestSpec);

    final String jobReportBucketArn = "arn:aws:s3:::ReportBucket";
    final String jobReportPrefix = "reports/governance-objects";
```

```
    final JobReport jobReport = new JobReport()
            .withEnabled(true)
            .withReportScope(JobReportScope.AllTasks)
            .withBucket(jobReportBucketArn)
            .withPrefix(jobReportPrefix)
            .withFormat(JobReportFormat.Report_CSV_20180820);

    final SimpleDateFormat format = new SimpleDateFormat("dd/MM/yyyy");
    final Date jan30th = format.parse("30/01/2025");

    final JobOperation jobOperation = new JobOperation()
            .withS3PutObjectRetention(new S3SetObjectRetentionOperation()
                    .withRetention(new S3Retention()
                            .withMode(S3ObjectLockRetentionMode.GOVERNANCE)
                            .withRetainUntilDate(jan30th)));

    final String roleArn = "arn:aws:iam::123456789012:role/bops-object-lock";
    final Boolean requiresConfirmation = true;
    final int priority = 10;

    final CreateJobRequest request = new CreateJobRequest()
            .withAccountId("123456789012")
            .withDescription("Put governance retention")
            .withManifest(manifestToPublicApi)
            .withOperation(jobOperation)
            .withPriority(priority)
            .withRoleArn(roleArn)
            .withReport(jobReport)
            .withConfirmationRequired(requiresConfirmation);

    final CreateJobResult result = awss3ControlClient.createJob(request);

    return result.getJobId();
}
```

Example Memintas tata kelola retensi pada beberapa objek

Contoh berikut dibangun berdasarkan contoh sebelumnya tentang pembuatan kebijakan kepercayaan, dan pengaturan izin konfigurasi Operasi Batch S3 serta Kunci Objek S3. Contoh ini menunjukkan cara memintas tata kelola retensi pada beberapa objek dan membuat pekerjaan Operasi Batch yang menggunakan bucket manifes dan melaporkan hasil di bucket laporan.

```
public void allowBypassGovernance() {
```

```
    final String roleName = "bops-object-lock";

    final String bypassGovernancePermissions = "{" +
            "    \"Version\": \"2012-10-17\"," +
            "    \"Statement\": [" +
            "        {" +
            "            \"Effect\": \"Allow\"," +
            "            \"Action\": [" +
            "                \"s3:BypassGovernanceRetention\"" +
            "            ]," +
            "            \"Resource\": [" +
            "                \"arn:aws:s3:::ManifestBucket/*\"" +
            "            ]" +
            "        }" +
            "    ]" +
            "}";

    final AmazonIdentityManagement iam =
            AmazonIdentityManagementClientBuilder.defaultClient();

    final PutRolePolicyRequest putRolePolicyRequest = new PutRolePolicyRequest()
            .withPolicyDocument(bypassGovernancePermissions)
            .withPolicyName("bypass-governance-permissions")
            .withRoleName(roleName);

    final PutRolePolicyResult putRolePolicyResult =
 iam.putRolePolicy(putRolePolicyRequest);
}
public String createRemoveGovernanceRetentionJob(final AWSS3ControlClient
 awss3ControlClient) {
    final String manifestObjectArn = "arn:aws:s3:::ManifestBucket/governance-objects-
manifest.csv";
    final String manifestObjectVersionId = "15ad5ba069e6bbc465c77bf83d541385";

    final JobManifestLocation manifestLocation = new JobManifestLocation()
            .withObjectArn(manifestObjectArn)
            .withETag(manifestObjectVersionId);

    final JobManifestSpec manifestSpec =
            new JobManifestSpec()
                    .withFormat(JobManifestFormat.S3BatchOperations_CSV_20180820)
                    .withFields("Bucket", "Key");

    final JobManifest manifestToPublicApi = new JobManifest()
```

```
            .withLocation(manifestLocation)
            .withSpec(manifestSpec);

    final String jobReportBucketArn = "arn:aws:s3:::ReportBucket";
    final String jobReportPrefix = "reports/bops-governance";

    final JobReport jobReport = new JobReport()
            .withEnabled(true)
            .withReportScope(JobReportScope.AllTasks)
            .withBucket(jobReportBucketArn)
            .withPrefix(jobReportPrefix)
            .withFormat(JobReportFormat.Report_CSV_20180820);

    final JobOperation jobOperation = new JobOperation()
            .withS3PutObjectRetention(new S3SetObjectRetentionOperation()
                    .withRetention(new S3Retention()));

    final String roleArn = "arn:aws:iam::123456789012:role/bops-object-lock";
    final Boolean requiresConfirmation = true;
    final int priority = 10;

    final CreateJobRequest request = new CreateJobRequest()
            .withAccountId("123456789012")
            .withDescription("Remove governance retention")
            .withManifest(manifestToPublicApi)
            .withOperation(jobOperation)
            .withPriority(priority)
            .withRoleArn(roleArn)
            .withReport(jobReport)
            .withConfirmationRequired(requiresConfirmation);

    final CreateJobResult result = awss3ControlClient.createJob(request);

    return result.getJobId();
}
```

Menggunakan Operasi Batch S3 untuk menonaktifkan penahanan hukum Kunci Objek S3

Contoh berikut dibangun berdasarkan contoh sebelumnya tentang pembuatan kebijakan kepercayaan, dan pengaturan izin konfigurasi Operasi Batch S3 serta Kunci Objek S3. Contoh ini

menunjukkan cara menonaktifkan penahanan hukum Kunci Objek pada objek menggunakan Operasi Batch.

Contoh ini pertama-tama akan memperbarui peran untuk memberikan izin `s3:PutObjectLegalHold`, membuat pekerjaan Operasi Batch yang menonaktifkan (menghapus) penahanan hukum dari objek yang diidentifikasi dalam manifes, kemudian melaporkan hal tersebut.

Menggunakan AWS CLI

Example Memperbarui peran untuk memberikan izin **`s3:PutObjectLegalHold`**

```
export AWS_PROFILE='aws-user'

read -d '' legal_hold_permissions <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObjectLegalHold"
            ],
            "Resource": [
                "arn:aws:s3:::ManifestBucket/*"
            ]
        }
    ]

EOF

aws iam put-role-policy --role-name bops-objectlock --policy-name legal-hold-
permissions --policy-document "${legal_hold_permissions}"
```

Example Menonaktifkan penahanan hukum

Contoh berikut menunjukkan cara menonaktifkan penahanan hukum.

```
export AWS_PROFILE='aws-user'
export AWS_DEFAULT_REGION='us-west-2'
export ACCOUNT_ID=123456789012
export ROLE_ARN='arn:aws:iam::123456789012:role/bops-objectlock'

read -d '' OPERATION <<EOF
```

```
{
  "S3PutObjectLegalHold": {
    "LegalHold": {
      "Status":"OFF"
    }
  }
}
EOF

read -d '' MANIFEST <<EOF
{
  "Spec": {
    "Format": "S3BatchOperations_CSV_20180820",
    "Fields": [
      "Bucket",
      "Key"
    ]
  },
  "Location": {
    "ObjectArn": "arn:aws:s3:::ManifestBucket/legalhold-object-manifest.csv",
    "ETag": "Your-manifest-ETag"
  }
}
EOF

read -d '' REPORT <<EOF
{
  "Bucket": "arn:aws:s3:::ReportBucket",
  "Format": "Report_CSV_20180820",
  "Enabled": true,
  "Prefix": "reports/legalhold-objects-bops",
  "ReportScope": "AllTasks"
}
EOF

aws \
    s3control create-job \
    --account-id "${ACCOUNT_ID}" \
    --manifest "${MANIFEST//$'\n'}" \
    --operation "${OPERATION//$'\n'/}" \
    --report "${REPORT//$'\n'}" \
    --priority 10 \
    --role-arn "${ROLE_ARN}" \
    --client-request-token "$(uuidgen)" \
```

```
    --region "${AWS_DEFAULT_REGION}" \
    --description "Turn off legal hold";
```

Menggunakan AWS SDK for Java

Example Memperbarui peran untuk memberikan izin **`s3:PutObjectLegalHold`**

```
public void allowPutObjectLegalHold() {
    final String roleName = "bops-object-lock";

    final String legalHoldPermissions = "{" +
            "    \"Version\": \"2012-10-17\"," +
            "    \"Statement\": [" +
            "        {" +
            "            \"Effect\": \"Allow\"," +
            "            \"Action\": [" +
            "                \"s3:PutObjectLegalHold\"" +
            "            ]," +
            "            \"Resource\": [" +
            "                \"arn:aws:s3:::ManifestBucket/*\"" +
            "            ]" +
            "        }" +
            "    ]" +
            "}";

    final AmazonIdentityManagement iam =
            AmazonIdentityManagementClientBuilder.defaultClient();

    final PutRolePolicyRequest putRolePolicyRequest = new PutRolePolicyRequest()
            .withPolicyDocument(legalHoldPermissions)
            .withPolicyName("legal-hold-permissions")
            .withRoleName(roleName);

    final PutRolePolicyResult putRolePolicyResult =
 iam.putRolePolicy(putRolePolicyRequest);
}
```

Example Menonaktifkan penahanan hukum

Gunakan contoh di bawah ini jika Anda ingin menonaktifkan penahanan hukum.

```
public String createLegalHoldOffJob(final AWSS3ControlClient awss3ControlClient) {
```

```
    final String manifestObjectArn = "arn:aws:s3:::ManifestBucket/legalhold-object-
manifest.csv";
    final String manifestObjectVersionId = "15ad5ba069e6bbc465c77bf83d541385";

    final JobManifestLocation manifestLocation = new JobManifestLocation()
            .withObjectArn(manifestObjectArn)
            .withETag(manifestObjectVersionId);

    final JobManifestSpec manifestSpec =
            new JobManifestSpec()
                    .withFormat(JobManifestFormat.S3BatchOperations_CSV_20180820)
                    .withFields("Bucket", "Key");

    final JobManifest manifestToPublicApi = new JobManifest()
            .withLocation(manifestLocation)
            .withSpec(manifestSpec);

    final String jobReportBucketArn = "arn:aws:s3:::ReportBucket";
    final String jobReportPrefix = "reports/legalhold-objects-bops";

    final JobReport jobReport = new JobReport()
            .withEnabled(true)
            .withReportScope(JobReportScope.AllTasks)
            .withBucket(jobReportBucketArn)
            .withPrefix(jobReportPrefix)
            .withFormat(JobReportFormat.Report_CSV_20180820);

    final JobOperation jobOperation = new JobOperation()
            .withS3PutObjectLegalHold(new S3SetObjectLegalHoldOperation()
                    .withLegalHold(new S3ObjectLockLegalHold()
                            .withStatus(S3ObjectLockLegalHoldStatus.OFF)));

    final String roleArn = "arn:aws:iam::123456789012:role/bops-object-lock";
    final Boolean requiresConfirmation = true;
    final int priority = 10;

    final CreateJobRequest request = new CreateJobRequest()
            .withAccountId("123456789012")
            .withDescription("Turn off legal hold")
            .withManifest(manifestToPublicApi)
            .withOperation(jobOperation)
            .withPriority(priority)
            .withRoleArn(roleArn)
            .withReport(jobReport)
```

```
            .withConfirmationRequired(requiresConfirmation);

    final CreateJobResult result = awss3ControlClient.createJob(request);

    return result.getJobId();
}
```

S3 Batch Operations

Tutorial berikut menyajikan end-to-end prosedur lengkap untuk beberapa tugas Operasi Batch.

- Tutorial: Video transcoding batch dengan Operasi Batch S3,, dan AWS LambdaAWS Elemental MediaConvert

Pemantauan Amazon S3

Pemantauan adalah bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja Amazon S3 dan solusi Anda AWS . Kami menyarankan untuk mengumpulkan data pemantauan dari semua bagian AWS solusi Anda sehingga Anda dapat lebih mudah men-debug kegagalan multipoint jika terjadi. Namun sebelum Anda mulai memantau Amazon S3, buat rencana pemantauan yang mencakup jawaban atas pertanyaan berikut:

- Apa tujuan pemantauan Anda?
- Sumber daya apa yang akan Anda pantau?
- Seberapa sering Anda akan memantau sumber daya ini?
- Alat pemantauan apa yang akan Anda gunakan?
- Siapa yang akan melakukan tugas pemantauan?
- Siapa yang harus diberi tahu saat terjadi kesalahan?

Untuk informasi lebih lanjut tentang pencatatan dan pemantauan di Amazon S3, lihat topik berikut.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Topik

- Alat pemantauan
- Opsi pencatatan untuk Amazon S3
- Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail
- Pencatatan permintaan dengan pencatatan akses server
- Memantau metrik dengan Amazon CloudWatch
- Notifikasi Peristiwa Amazon S3

Alat pemantauan

AWS menyediakan berbagai alat yang dapat Anda gunakan untuk memantau Amazon S3. Anda dapat mengonfigurasi beberapa alat ini untuk melakukan pemantauan untuk Anda, sementara beberapa alat memerlukan intervensi manual. Kami menyarankan agar Anda mengautomasi tugas pemantauan sebanyak mungkin.

Alat pemantauan otomatis

Anda dapat menggunakan alat pemantauan otomatis berikut untuk melihat Amazon S3 dan melaporkan saat terjadi kesalahan:

- CloudWatch Alarm Amazon — Tonton satu metrik selama periode waktu yang Anda tentukan, dan lakukan satu atau beberapa tindakan berdasarkan nilai metrik relatif terhadap ambang batas tertentu selama beberapa periode waktu. Tindakannya adalah pemberitahuan yang dikirim ke topik Amazon Simple Notification Service (Amazon SNS) atau kebijakan Amazon EC2 Auto Scaling. CloudWatch alarm tidak memanggil tindakan hanya karena mereka berada dalam keadaan tertentu. Status harus diubah dan dipelihara selama jangka waktu tertentu. Untuk informasi selengkapnya, lihat Memantau metrik dengan Amazon CloudWatch.
- AWS CloudTrail Pemantauan Log - Bagikan file log antar akun, pantau file CloudTrail log secara real time dengan mengirimkannya ke CloudWatch Log, menulis aplikasi pemrosesan log di Java, dan validasi bahwa file log Anda tidak berubah setelah pengiriman oleh CloudTrail. Untuk informasi selengkapnya, lihat Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail.

Alat pemantauan manual

Bagian penting lainnya dari pemantauan Amazon S3 melibatkan pemantauan secara manual item-item yang tidak CloudWatch tercakup oleh alarm. Amazon S3,, CloudWatch Trusted Advisor, dan AWS Management Console dasbor lainnya memberikan at-a-glance tampilan keadaan lingkungan Anda. AWS Anda mungkin ingin mengaktifkan pencatatan akses server, yang melacak permintaan akses ke bucket Anda. Setiap catatan log akses memberikan perincian tentang permintaan akses tunggal, seperti pemohon, nama bucket, waktu permintaan, tindakan permintaan, status respons, dan kode kesalahan, jika ada. Untuk informasi selengkapnya, lihat Pencatatan permintaan dengan pencatatan akses server.

- Dasbor Amazon S3 menunjukkan hal berikut:
  - Bucket Anda dan objek serta properti yang dimuat di dalamnya

- CloudWatch Halaman beranda menunjukkan yang berikut:
  - Alarm dan status saat ini
  - Grafik alarm dan sumber daya
  - Status kesehatan layanan

  Selain itu, Anda dapat menggunakan CloudWatch untuk melakukan hal berikut:
  - Buat dasbor yang disesuaikan untuk memantau layanan yang Anda pedulikan.
  - Data metrik grafik untuk memecahkan masalah dan menemukan tren.
  - Cari dan telusuri semua metrik AWS sumber daya Anda.
  - Buat dan sunting alarm untuk menerima pemberitahuan tentang masalah.
- AWS Trusted Advisor dapat membantu Anda memantau AWS sumber daya Anda untuk meningkatkan kinerja, keandalan, keamanan, dan efektivitas biaya. Empat pemeriksaan Trusted Advisor tersedia bagi semua pengguna; lebih dari 50 pemeriksaan tersedia bagi pengguna dengan perencanaan dukungan Bisnis atau Korporasi. Untuk informasi selengkapnya, lihat AWS Trusted Advisor.

  Trusted Advisor memiliki pemeriksaan ini yang berhubungan dengan Amazon S3:
  - Pemeriksaan konfigurasi pencatatan log bucket Amazon S3.
  - Pemeriksaan keamanan untuk bucket Amazon S3 yang memiliki izin akses terbuka.
  - Pengecekan toleransi kesalahan untuk bucket Amazon S3 yang tidak memiliki Penentuan Versi yang diaktifkan, atau memiliki Penentuan Versi yang ditangguhkan.

Opsi pencatatan untuk Amazon S3

Anda dapat merekam tindakan yang diambil oleh pengguna, peran, atau sumber daya Layanan AWS Amazon S3 dan memelihara catatan log untuk tujuan audit dan kepatuhan. Untuk melakukan ini, Anda dapat menggunakan pencatatan akses server, pencatatan AWS CloudTrail , atau kombinasi keduanya. Sebaiknya gunakan CloudTrail untuk mencatat tindakan tingkat ember dan tingkat objek untuk sumber daya Amazon S3 Anda. Untuk informasi selengkapnya tentang setiap opsi, lihat bagian berikut:

- Pencatatan permintaan dengan pencatatan akses server
- Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail

Tabel berikut mencantumkan properti kunci CloudTrail log dan log akses server Amazon S3. Untuk memastikan bahwa CloudTrail memenuhi persyaratan keamanan Anda, tinjau tabel dan catatan.

Properti log	AWS CloudTrail	Log server Amazon S3
Dapat diteruskan ke sistem lain (Amazon CloudWatch Log, Amazon Events) CloudWatch	Ya	Tidak
Kirim log ke lebih dari satu tujuan (misalnya, kirim log yang sama ke dua bucket yang berbeda)	Ya	Tidak
Mengaktifkan log untuk subset objek (prefiks)	Ya	Tidak
Pengiriman log akun silang (bucket target dan sumber yang dimiliki oleh akun yang berbeda)	Ya	Tidak
Validasi integritas file log dengan menggunakan tanda tangan digital atau hashing	Ya	Tidak
Default atau pilihan enkripsi untuk file log	Ya	Tidak
Operasi objek (dengan menggunakan API Amazon S3)	Ya	Ya
Operasi bucket (dengan menggunakan API Amazon S3)	Ya	Ya
UI yang dapat dicari untuk log	Ya	Tidak

Properti log	AWS CloudTrail	Log server Amazon S3
Bidang untuk parameter Kunci Objek, Amazon S3 Pilih properti untuk catatan log	Ya	Tidak
Bidang untuk `Object Size`, `Total Time`, `Turn-Around Time` , dan `HTTP Referer` untuk catatan log	Tidak	Ya
Transisi Lifecycle, kedaluwarsa, pemulihan	Tidak	Ya
Pencatatan log kunci dalam operasi penghapusan batch	Tidak	Ya
Kegagalan autentikasi[1]	Tidak	Ya
Akun tempat log dikirimkan	Pemilik bucket[2], dan pemohon	Hanya pemilik bucket
Performance and Cost	AWS CloudTrail	Amazon S3 Server Logs
Harga	Peristiwa manajemen (pengiriman pertama) gratis; peristiwa data dikenakan biaya, selain penyimpanan log	Tidak ada biaya lain selain penyimpanan log
Kecepatan pengiriman log	Peristiwa data setiap 5 menit; peristiwa manajemen setiap 15 menit	Dalam beberapa jam
Format log	JSON	Berkas log dengan catatan yang dipisahkan oleh ruang, yang tidak terbatas pada baris baru

Catatan

1. CloudTrail tidak mengirimkan log untuk permintaan yang gagal otentikasi (di mana kredensyal yang diberikan tidak valid). Namun demikian, hal tersebut mencakup log untuk permintaan yang gagal diterima otorisasi (`AccessDenied`) dan permintaan yang dibuat oleh pengguna anonim.
2. Pemilik bucket S3 menerima CloudTrail log ketika akun tidak memiliki akses penuh ke objek dalam permintaan. Untuk informasi selengkapnya, lihat Tindakan tingkat objek Amazon S3 dalam skenario lintas akun.
3. S3 tidak mendukung pengiriman CloudTrail log atau log akses server ke pemohon atau pemilik bucket untuk permintaan titik akhir VPC saat kebijakan titik akhir VPC menyangkalnya atau untuk permintaan yang gagal sebelum kebijakan VPC dievaluasi.

Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail

Amazon S3 terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau. Layanan AWS CloudTrail menangkap semua panggilan API untuk Amazon S3 sebagai peristiwa. Panggilan yang diambil termasuk panggilan dari konsol Amazon S3 dan panggilan kode ke operasi API Amazon S3. Dengan menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat ke Amazon S3, alamat IP dari mana permintaan dibuat, kapan dibuat, dan detail tambahan.

Setiap entri peristiwa atau log berisi informasi tentang siapa yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan berikut hal ini:

- Baik permintaan tersebut dibuat dengan kredensial pengguna root atau pengguna.
- Apakah permintaan dibuat atas nama pengguna IAM Identity Center.
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna terfederasi.
- Apakah permintaan tersebut dibuat oleh Layanan AWS lain.

CloudTrail aktif di Anda Akun AWS ketika Anda membuat akun dan Anda secara otomatis memiliki akses ke riwayat CloudTrail Acara. Riwayat CloudTrail Acara menyediakan catatan yang dapat dilihat, dapat dicari, dapat diunduh, dan tidak dapat diubah dari 90 hari terakhir dari peristiwa manajemen yang direkam dalam file. Wilayah AWS Untuk informasi selengkapnya, lihat Bekerja

dengan riwayat CloudTrail Acara di Panduan AWS CloudTrail Pengguna. Tidak ada CloudTrail biaya untuk melihat riwayat Acara.

Untuk catatan acara yang sedang berlangsung dalam 90 hari Akun AWS terakhir Anda, buat jejak atau penyimpanan data acara CloudTrailDanau.

CloudTrail jalan setapak

Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Semua jalur yang dibuat menggunakan AWS Management Console Multi-region. Anda dapat membuat jalur Single-region atau Multi-region dengan menggunakan. AWS CLI Membuat jejak Multi-wilayah disarankan karena Anda menangkap aktivitas Wilayah AWS di semua akun Anda. Jika Anda membuat jejak wilayah Tunggal, Anda hanya dapat melihat peristiwa yang dicatat di jejak. Wilayah AWS Untuk informasi selengkapnya tentang jejak, lihat Membuat jejak untuk Anda Akun AWS dan Membuat jejak untuk organisasi di Panduan AWS CloudTrail Pengguna.

Anda dapat mengirimkan satu salinan acara manajemen yang sedang berlangsung ke bucket Amazon S3 Anda tanpa biaya CloudTrail dengan membuat jejak, namun, ada biaya penyimpanan Amazon S3. Untuk informasi selengkapnya tentang CloudTrail harga, lihat AWS CloudTrail Harga. Untuk informasi tentang harga Amazon S3, lihat Harga Amazon S3.

CloudTrail Menyimpan data acara danau

CloudTrail Lake memungkinkan Anda menjalankan kueri berbasis SQL pada acara Anda. CloudTrail Lake mengonversi peristiwa yang ada dalam format JSON berbasis baris ke format Apache ORC. ORC adalah format penyimpanan kolumnar yang dioptimalkan untuk pengambilan data dengan cepat. Peristiwa digabungkan ke dalam penyimpanan data peristiwa, yang merupakan kumpulan peristiwa yang tidak dapat diubah berdasarkan kriteria yang Anda pilih dengan menerapkan pemilih acara tingkat lanjut. Penyeleksi yang Anda terapkan ke penyimpanan data acara mengontrol peristiwa mana yang bertahan dan tersedia untuk Anda kueri. Untuk informasi lebih lanjut tentang CloudTrail Danau, lihat Bekerja dengan AWS CloudTrail Danau di Panduan AWS CloudTrail Pengguna.

CloudTrail Penyimpanan data acara danau dan kueri menimbulkan biaya. Saat Anda membuat penyimpanan data acara, Anda memilih opsi harga yang ingin Anda gunakan untuk penyimpanan data acara. Opsi penetapan harga menentukan biaya untuk menelan dan menyimpan peristiwa, dan periode retensi default dan maksimum untuk penyimpanan data acara. Untuk informasi selengkapnya tentang CloudTrail harga, lihat AWS CloudTrail Harga.

Anda dapat menyimpan file log dalam bucket selama yang Anda inginkan, tetapi Anda juga dapat menentukan aturan siklus hidup Amazon S3 untuk mengarsipkan atau menghapus file log secara otomatis. Secara default, file log Anda dienkripsi dengan menggunakan enkripsi di sisi server (SSE) Amazon S3.

Menggunakan CloudTrail log dengan log akses server Amazon S3 dan Log CloudWatch

AWS CloudTrail log menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di Amazon S3, sementara log akses server Amazon S3 menyediakan catatan terperinci untuk permintaan yang dibuat ke bucket S3. Untuk informasi lebih lanjut tentang cara kerja pencatatan yang berbeda, dan properti, performa, dan biayanya, lihat the section called “Opsi pencatatan”.

Anda dapat menggunakan AWS CloudTrail log bersama dengan log akses server untuk Amazon S3. CloudTrail log memberi Anda pelacakan API terperinci untuk operasi tingkat ember dan tingkat objek Amazon S3. Log akses server untuk Amazon S3 memberi Anda visibilitas ke operasi tingkat objek pada data Anda di Amazon S3. Untuk informasi lebih lanjut tentang log akses server, lihat Pencatatan permintaan dengan pencatatan akses server.

Anda juga dapat menggunakan CloudTrail log bersama dengan Amazon CloudWatch untuk Amazon S3. CloudTrail integrasi dengan CloudWatch Log memberikan aktivitas API tingkat ember S3 yang ditangkap oleh aliran CloudWatch log di CloudTrail grup log yang Anda tentukan CloudWatch . Anda dapat membuat CloudWatch alarm untuk memantau aktivitas API tertentu dan menerima pemberitahuan email saat aktivitas API tertentu terjadi. Untuk informasi selengkapnya tentang CloudWatch alarm untuk memantau aktivitas API tertentu, lihat Panduan AWS CloudTrail Pengguna. Untuk informasi selengkapnya tentang penggunaan CloudWatch dengan Amazon S3, lihat. Memantau metrik dengan Amazon CloudWatch

> **Note**
>
> S3 tidak mendukung pengiriman CloudTrail log ke pemohon atau pemilik bucket untuk permintaan titik akhir VPC saat kebijakan titik akhir VPC menolaknya.

CloudTrail melacak dengan panggilan API SOAP Amazon S3

CloudTrail melacak panggilan API SOAP Amazon S3. Dukungan SOAP Amazon S3 melalui HTTP dihentikan, tetapi masih tersedia melalui HTTPS. Untuk informasi lebih lanjut tentang dukungan SOAP Amazon S3, lihat Lampiran a: Menggunakan SOAP API.

> **Important**
>
> Fitur Amazon S3 baru tidak didukung untuk SOAP. Kami menyarankan Anda menggunakan REST API atau AWS SDK.

Tindakan SOAP Amazon S3 dilacak dengan logging CloudTrail

Nama SOAP API	Nama peristiwa API yang digunakan di CloudTrail log
ListAllMyBuckets	`ListBuckets`
CreateBucket	`CreateBucket`
DeleteBucket	`DeleteBucket`
GetBucketAccessControlPolicy	`GetBucketAcl`
SetBucketAccessControlPolicy	`PutBucketAcl`
GetBucketLoggingStatus	`GetBucketLogging`
SetBucketLoggingStatus	`PutBucketLogging`

Untuk informasi selengkapnya tentang CloudTrail Amazon S3, lihat topik berikut:

Topik

- Acara Amazon S3 CloudTrail
- CloudTrail entri file log untuk Amazon S3 dan S3 di Outposts
- Mengaktifkan pencatatan CloudTrail peristiwa untuk bucket dan objek S3
- Mengidentifikasi permintaan Amazon S3 menggunakan CloudTrail

Acara Amazon S3 CloudTrail

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Bagian ini memberikan informasi tentang peristiwa yang dicatat oleh S3. CloudTrail

Peristiwa data Amazon S3 di CloudTrail

Peristiwa data memberikan informasi tentang operasi sumber daya yang dilakukan pada atau di sumber daya (misalnya, membaca atau menulis ke objek Amazon S3). Ini juga dikenal sebagai operasi bidang data. Peristiwa data seringkali merupakan aktivitas volume tinggi. Secara default, CloudTrail tidak mencatat peristiwa data. Riwayat CloudTrail peristiwa tidak merekam peristiwa data.

Biaya tambahan berlaku untuk peristiwa data. Untuk informasi selengkapnya tentang CloudTrail harga, lihat AWS CloudTrail Harga.

Anda dapat mencatat peristiwa data untuk jenis sumber daya Amazon S3 menggunakan CloudTrail konsol AWS CLI, atau operasi CloudTrail API. Untuk informasi selengkapnya tentang cara mencatat peristiwa data, lihat Mencatat peristiwa data dengan AWS Management Console dan Logging peristiwa data dengan AWS Command Line Interface di Panduan AWS CloudTrail Pengguna.

Tabel berikut mencantumkan jenis sumber daya Amazon S3 yang dapat Anda log peristiwa data. Kolom tipe peristiwa data (konsol) menunjukkan nilai yang akan dipilih dari daftar tipe peristiwa Data di CloudTrail konsol. Kolom nilai resources.type menunjukkan **`resources.type`** nilai, yang akan Anda tentukan saat mengonfigurasi penyeleksi acara lanjutan menggunakan API atau. AWS CLI CloudTrail CloudTrailKolom API Data yang dicatat ke menampilkan panggilan API yang dicatat CloudTrail untuk jenis sumber daya.

Jenis peristiwa data (konsol)	nilai resources.type	API data masuk CloudTrail
S3	`AWS::S3::Object`	• AbortMultipartUpload • CompleteMultipartUpload • CopyObject • CreateMultipartUpload • DeleteObject • DeleteObjectTagging • DeleteObjects • GetObject • GetObjectAcl • GetObjectAttributes • GetObjectLegalHold • GetObjectRetention • GetObjectTagging • GetObjectTorrent • HeadObject • ListMultipartUploads • ListObjectVersions • ListObjects • ListParts • PutObject • PutObjectAcl • PutObjectLegalHold • PutObjectRetention • PutObjectTagging • RestoreObject • SelectObjectContent • UploadPart • UploadPartCopy

Jenis peristiwa data (konsol)	nilai resources.type	API data masuk CloudTrail
Titik Akses S3	`AWS::S3::Access Point`	• AbortMultipartUpload • CompleteMultipartUpload • CopyObject (hanya salinan wilayah yang sama) • CreateMultipartUpload • DeleteObject • DeleteObjectTagging • GetBucketAcl • GetBucketCors • GetBucketLocation • GetBucketNotificationConfiguration • GetBucketPolicy • GetObject • GetObjectAcl • GetObjectAttributes • GetObjectLegalHold • GetObjectRetention • GetObjectTagging • HeadBucket • HeadObject • ListMultipartUploads • ListObjects • ListObjectsV2 • ListObjectVersions • ListParts • Presign • PutObject • PutObjectLegalHold

Jenis peristiwa data (konsol)	nilai resources.type	API data masuk CloudTrail
		• PutObjectRetention • PutObjectAcl • PutObjectTagging • RestoreObject • UploadPart • UploadPartCopy (hanya salinan wilayah yang sama)

Jenis peristiwa data (konsol)	nilai resources.type	API data masuk CloudTrail
S3 Object Lambda	`AWS::S3ObjectLambda::AccessPoint`	• AbortMultipartUpload • CompleteMultipartUpload • CopyObject (hanya salinan wilayah yang sama) • CreateMultipartUpload • DeleteObject • DeleteObjectTagging • GetObject • GetObjectAcl • GetObjectLegalHold • GetObjectRetention • GetObjectTagging • HeadObject • ListMultipartUploads • ListObjects • ListObjectVersions • ListParts • PutObject • PutObjectLegalHold • PutObjectRetention • PutObjectAcl • PutObjectTagging • RestoreObject • UploadPart • WriteGetObjectResponse

Jenis peristiwa data (konsol)	nilai resources.type	API data masuk CloudTrail
Outposts S3	`AWS::S3Outposts::Object`	• AbortMultipartUpload • CompleteMultipartUpload • CopyObject (hanya salinan wilayah yang sama) • CreateMultipartUpload • DeleteObject • DeleteObjectTagging • GetObject • GetObjectTagging • HeadObject • ListMultipartUploads • ListObjects • ListObjectsV2 • ListParts • PutObject • PutObjectTagging • UploadPart • UploadPartCopy

Anda dapat mengonfigurasi pemilih acara lanjutan untuk memfilter pada `eventName``readOnly`,, dan `resources.ARN` bidang untuk mencatat hanya peristiwa yang penting bagi Anda. Untuk informasi selengkapnya tentang bidang ini, lihat AdvancedFieldSelectordi Referensi AWS CloudTrail API.

Acara manajemen Amazon S3 di CloudTrail

Amazon S3 mencatat semua operasi pesawat kontrol sebagai peristiwa manajemen. Untuk informasi selengkapnya tentang operasi API S3, lihat Referensi API Amazon S3.

Cara CloudTrail menangkap permintaan yang dibuat ke Amazon S3

Secara default, CloudTrail mencatat panggilan API tingkat ember S3 yang dibuat dalam 90 hari terakhir, tetapi tidak mencatat permintaan yang dibuat ke objek. Panggilan tingkat bucket mencakup

peristiwa seperti `CreateBucket`, `DeleteBucket`, `PutBucketLifecycle`, `PutBucketPolicy`, dan seterusnya. Anda dapat melihat peristiwa tingkat ember di konsol. CloudTrail Namun, Anda tidak dapat melihat peristiwa data (panggilan tingkat objek Amazon S3) di sana—Anda harus mengurai atau menanyakan log untuknya. CloudTrail

Tindakan tingkat akun Amazon S3 dilacak dengan pencatatan CloudTrail

CloudTrail mencatat tindakan tingkat akun. Catatan Amazon S3 ditulis bersama dengan Layanan AWS catatan lain dalam file log. CloudTrail menentukan kapan harus membuat dan menulis ke file baru berdasarkan periode waktu dan ukuran file.

Tabel di bagian ini mencantumkan tindakan tingkat akun Amazon S3 yang didukung untuk pencatatan. CloudTrail

Tindakan API tingkat akun Amazon S3 yang dilacak dengan CloudTrail logging muncul sebagai nama peristiwa berikut. Nama CloudTrail acara berbeda dari nama tindakan API. Sebagai contoh, DeletePublicAccessBlock adalah DeleteAccountPublicAccessBlock.

- DeleteAccountPublicAccessBlock
- GetAccountPublicAccessBlock
- PutAccountPublicAccessBlock

Tindakan tingkat ember Amazon S3 yang dilacak dengan logging CloudTrail

Secara default, CloudTrail mencatat tindakan tingkat ember untuk bucket tujuan umum. Catatan Amazon S3 ditulis bersama dengan catatan AWS layanan lain dalam file log. CloudTrail menentukan kapan harus membuat dan menulis ke file baru berdasarkan periode waktu dan ukuran file.

Bagian ini mencantumkan tindakan tingkat ember Amazon S3 yang didukung untuk pencatatan oleh. CloudTrail

Tindakan API tingkat ember Amazon S3 yang dilacak dengan CloudTrail logging muncul sebagai nama peristiwa berikut. Dalam beberapa kasus, nama CloudTrail acara berbeda dari nama tindakan API. Sebagai contoh, `PutBucketLifecycleConfiguration` adalah`PutBucketLifecycle`.

- CreateBucket
- DeleteBucket
- DeleteBucketAnalyticsConfiguration

- DeleteBucketCors
- DeleteBucketEncryption
- DeleteBucketIntelligentTieringConfiguration
- DeleteBucketInventoryConfiguration
- DeleteBucketLifecycle
- DeleteBucketMetricsConfiguration
- DeleteBucketOwnershipControls
- DeleteBucketPolicy
- DeleteBucketPublicAccessBlock
- DeleteBucketReplication
- DeleteBucketTagging
- GetAccelerateConfiguration
- GetBucketAcl
- GetBucketAnalyticsConfiguration
- GetBucketCors
- GetBucketEncryption
- GetBucketIntelligentTieringConfiguration
- GetBucketInventoryConfiguration
- GetBucketLifecycle
- GetBucketLocation
- GetBucketLogging
- GetBucketMetricsConfiguration
- GetBucketNotification
- GetBucketObjectLockConfiguration
- GetBucketOwnershipControls
- GetBucketPolicy
- GetBucketPolicyStatus
- GetBucketPublicAccessBlock
- GetBucketReplication

- GetBucketRequestPayment
- GetBucketTagging
- GetBucketVersioning
- GetBucketWebsite
- HeadBucket
- ListBuckets
- PutAccelerateConfiguration
- PutBucketAcl
- PutBucketAnalyticsConfiguration
- PutBucketCors
- PutBucketEncryption
- PutBucketIntelligentTieringConfiguration
- PutBucketInventoryConfiguration
- PutBucketLifecycle
- PutBucketLogging
- PutBucketMetricsConfiguration
- PutBucketNotification
- PutBucketObjectLockConfiguration
- PutBucketOwnershipControls
- PutBucketPolicy
- PutBucketPublicAccessBlock
- PutBucketReplication
- PutBucketRequestPayment
- PutBucketTagging
- PutBucketVersioning
- PutBucketWebsite

Selain operasi API ini, Anda juga dapat menggunakan tindakan tingkat objek OPTIONS objek. Tindakan ini diperlakukan seperti tindakan tingkat ember dalam CloudTrail logging karena tindakan memeriksa konfigurasi CORS bucket.

Tindakan tingkat ember S3 Express One Zone (titik akhir API Regional) yang dilacak dengan pencatatan CloudTrail

Secara default, CloudTrail mencatat tindakan tingkat ember untuk bucket direktori sebagai peristiwa manajemen. Acara `eventsource` untuk CloudTrail manajemen untuk S3 Express One Zone adalah`s3express.amazonaws.com`.

> **Note**
>
> Untuk S3 Express One Zone, CloudTrail pencatatan operasi API titik akhir Zonal (level objek, atau bidang data) (misalnya, `PutObject` atau`GetObject`) tidak didukung.

Operasi API titik akhir Regional berikut ini dicatat. CloudTrail

- CreateBucket
- DeleteBucket
- DeleteBucketPolicy
- GetBucketPolicy
- PutBucketPolicy
- ListDirectoryBuckets

Untuk informasi selengkapnya, lihat Praktik terbaik keamanan untuk S3 Express One Zone.

Tindakan tingkat objek Amazon S3 dalam skenario lintas akun

Berikut ini adalah kasus penggunaan khusus yang melibatkan panggilan API tingkat objek dalam skenario lintas akun dan bagaimana CloudTrail log dilaporkan. CloudTrail mengirimkan log ke pemohon (akun yang melakukan panggilan API), kecuali dalam beberapa kasus akses ditolak di mana entri log disunting atau dihilangkan. Saat mengatur akses lintas akun, pertimbangkan contoh di bagian ini.

> **Note**
>
> Contoh mengasumsikan bahwa CloudTrail log dikonfigurasi dengan tepat.

Contoh 1: CloudTrail mengirimkan log ke pemilik bucket

CloudTrail mengirimkan log ke pemilik bucket meskipun pemilik bucket tidak memiliki izin untuk operasi API objek yang sama. Pertimbangkan skenario lintas akun berikut ini:

- Akun A adalah pemilik bucket.
- Akun-B (peminta) mencoba mengakses objek dalam bucket tersebut.
- Akun-C memiliki objek. Akun C mungkin atau mungkin bukan akun yang sama dengan Akun A.

> **Note**
>
> CloudTrail selalu mengirimkan log API tingkat objek ke pemohon (Akun B). Selain itu, CloudTrail juga mengirimkan log yang sama ke pemilik bucket (Akun A) bahkan ketika pemilik bucket tidak memiliki objek (Akun C) atau memiliki izin untuk operasi API yang sama pada objek tersebut.

Contoh 2: CloudTrail tidak memperbanyak alamat email yang digunakan dalam pengaturan ACL objek

Pertimbangkan skenario lintas akun berikut ini:

- Akun A adalah pemilik bucket.
- Akun B (pemohon) mengirimkan permintaan untuk menetapkan pemberian izin ACL objek dengan menggunakan alamat surel. Untuk informasi selengkapnya tentang ACL, lihat Gambaran umum daftar kontrol akses (ACL).

Pemohon mendapatkan log beserta informasi surel. Namun, pemilik bucket — jika mereka memenuhi syarat untuk menerima log, seperti pada contoh 1—mendapatkan log yang melaporkan peristiwa tersebut CloudTrail . Namun, pemilik bucket tidak mendapatkan konfigurasi ACL, kustomnya alamat email penerima izin dan izinnya. Satu-satunya informasi yang diberitahukan oleh log kepada pemilik bucket adalah bahwa panggilan API ACL dilakukan oleh Akun-B.

CloudTrail entri file log untuk Amazon S3 dan S3 di Outposts

> **Important**
>
> Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis akan dienkripsi tanpa biaya tambahan dan tidak akan berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Peristiwa mewakili permintaan tunggal dari sumber manapun dan mencakup informasi tentang operasi API yang diminta, tanggal dan waktu operasi, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, sehingga peristiwa tidak muncul dalam urutan tertentu.

Untuk informasi selengkapnya, lihat contoh berikut ini.

Topik

- Contoh: entri file CloudTrail log untuk Amazon S3
- Contoh: entri file log Outposts di Amazon S3

Contoh: entri file CloudTrail log untuk Amazon S3

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan GETLayanan, PutBucketAcl, dan GetBucketVersioningtindakan.

```
{
    "Records": [
    {
        "eventVersion": "1.03",
        "userIdentity": {
            "type": "IAMUser",
            "principalId": "111122223333",
            "arn": "arn:aws:iam::111122223333:user/myUserName",
            "accountId": "111122223333",
```

```
            "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
            "userName": "myUserName"
        },
        "eventTime": "2019-02-01T03:18:19Z",
        "eventSource": "s3.amazonaws.com",
        "eventName": "ListBuckets",
        "awsRegion": "us-west-2",
        "sourceIPAddress": "127.0.0.1",
        "userAgent": "[]",
        "requestParameters": {
            "host": [
                "s3.us-west-2.amazonaws.com"
            ]
        },
        "responseElements": null,
        "additionalEventData": {
            "SignatureVersion": "SigV2",
            "AuthenticationMethod": "QueryString",
            "aclRequired": "Yes"
    },
        "requestID": "47B8E8D397DCE7A6",
        "eventID": "cdc4b7ed-e171-4cef-975a-ad829d4123e8",
        "eventType": "AwsApiCall",
        "recipientAccountId": "444455556666",
        "tlsDetails": {
            "tlsVersion": "TLSv1.2",
            "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
            "clientProvidedHostHeader": "s3.amazonaws.com"
    }
    },
    {
       "eventVersion": "1.03",
       "userIdentity": {
            "type": "IAMUser",
            "principalId": "111122223333",
            "arn": "arn:aws:iam::111122223333:user/myUserName",
            "accountId": "111122223333",
            "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
            "userName": "myUserName"
        },
      "eventTime": "2019-02-01T03:22:33Z",
      "eventSource": "s3.amazonaws.com",
      "eventName": "PutBucketAcl",
      "awsRegion": "us-west-2",
```

```
      "sourceIPAddress": "",
      "userAgent": "[]",
      "requestParameters": {
          "bucketName": "",
          "AccessControlPolicy": {
              "AccessControlList": {
                  "Grant": {
                      "Grantee": {
                          "xsi:type": "CanonicalUser",
                          "xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",
                          "ID":
 "d25639fbe9c19cd30a4c0f43fbf00e2d3f96400a9aa8dabfbbebe1906Example"
                       },
                      "Permission": "FULL_CONTROL"
                   }
              },
              "xmlns": "http://s3.amazonaws.com/doc/2006-03-01/",
              "Owner": {
                  "ID":
 "d25639fbe9c19cd30a4c0f43fbf00e2d3f96400a9aa8dabfbbebe1906Example"
              }
          },
          "host": [
              "s3.us-west-2.amazonaws.com"
          ],
          "acl": [
              ""
          ]
      },
      "responseElements": null,
      "additionalEventData": {
          "SignatureVersion": "SigV4",
          "CipherSuite": "ECDHE-RSA-AES128-SHA",
          "AuthenticationMethod": "AuthHeader"
      },
      "requestID": "BD8798EACDD16751",
      "eventID": "607b9532-1423-41c7-b048-ec2641693c47",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333",
      "tlsDetails": {
            "tlsVersion": "TLSv1.2",
            "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
            "clientProvidedHostHeader": "s3.amazonaws.com"
    }
```

```
    },
    {
      "eventVersion": "1.03",
      "userIdentity": {
          "type": "IAMUser",
          "principalId": "111122223333",
          "arn": "arn:aws:iam::111122223333:user/myUserName",
          "accountId": "111122223333",
          "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
          "userName": "myUserName"
        },
      "eventTime": "2019-02-01T03:26:37Z",
      "eventSource": "s3.amazonaws.com",
      "eventName": "GetBucketVersioning",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "",
      "userAgent": "[]",
      "requestParameters": {
          "host": [
              "s3.us-west-2.amazonaws.com"
          ],
          "bucketName": "example-s3-bucket1",
          "versioning": [
              ""
          ]
      },
      "responseElements": null,
      "additionalEventData": {
          "SignatureVersion": "SigV4",
          "CipherSuite": "ECDHE-RSA-AES128-SHA",
          "AuthenticationMethod": "AuthHeader"
    },
      "requestID": "07D681279BD94AED",
      "eventID": "f2b287f3-0df1-4961-a2f4-c4bdfed47657",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333",
      "tlsDetails": {
            "tlsVersion": "TLSv1.2",
            "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
            "clientProvidedHostHeader": "s3.amazonaws.com"
    }
    }
  ]
```

```
}
```

Contoh: entri file log Outposts di Amazon S3

Amazon S3 pada acara manajemen Outposts tersedia melalui. AWS CloudTrail Untuk informasi selengkapnya, lihat Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail. Selain itu, Anda dapat secara opsional mengaktifkan pencatatan untuk peristiwa data di AWS CloudTrail.

Jejak adalah konfigurasi yang mengaktifkan pengiriman peristiwa sebagai file log ke bucket S3 Amazon di Wilayah yang Anda tentukan. CloudTrail log untuk bucket Outposts Anda menyertakan bidang baru`edgeDeviceDetails`, yang mengidentifikasi Outpost tempat bucket yang ditentukan berada.

Bidang log tambahan mencakup tindakan yang diminta, tanggal dan waktu tindakan, dan parameter permintaan. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, sehingga file tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan PutObjecttindakan pada`s3-outposts`.

```
{
      "eventVersion": "1.08",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "111122223333",
        "arn": "arn:aws:iam::111122223333:user/yourUserName",
        "accountId": "222222222222",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "yourUserName"
      },
      "eventTime": "2020-11-30T15:44:33Z",
      "eventSource": "s3-outposts.amazonaws.com",
      "eventName": "PutObject",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "26.29.66.20",
      "userAgent": "aws-cli/1.18.39 Python/3.4.10 Darwin/18.7.0 botocore/1.15.39",
      "requestParameters": {
        "expires": "Wed, 21 Oct 2020 07:28:00 GMT",
        "Content-Language": "english",
        "x-amz-server-side-encryption-customer-key-MD5": "wJalrXUtnFEMI/K7MDENG/
bPxRfiCYEXAMPLEKEY",
        "ObjectCannedACL": "BucketOwnerFullControl",
```

```
        "x-amz-server-side-encryption": "Aes256",
        "Content-Encoding": "gzip",
        "Content-Length": "10",
        "Cache-Control": "no-cache",
        "Content-Type": "text/html; charset=UTF-8",
        "Content-Disposition": "attachment",
        "Content-MD5": "je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY",
        "x-amz-storage-class": "Outposts",
        "x-amz-server-side-encryption-customer-algorithm": "Aes256",
        "bucketName": "example-s3-bucket1",
        "Key": "path/upload.sh"
      },
      "responseElements": {
        "x-amz-server-side-encryption-customer-key-MD5": "wJalrXUtnFEMI/K7MDENG/
bPxRfiCYEXAMPLEKEY",
        "x-amz-server-side-encryption": "Aes256",
        "x-amz-version-id": "001",
        "x-amz-server-side-encryption-customer-algorithm": "Aes256",
        "ETag": "d41d8cd98f00b204e9800998ecf8427f"
      },
      "additionalEventData": {
        "CipherSuite": "ECDHE-RSA-AES128-SHA",
        "bytesTransferredIn": 10,
        "x-amz-id-2": "29xXQBV20
+xOHKItvzY1suLv1i6A52E0zOX159fpfsItYd58JhXwKxXAXI4IQkp6",
        "SignatureVersion": "SigV4",
        "bytesTransferredOut": 20,
        "AuthenticationMethod": "AuthHeader"
      },
      "requestID": "8E96D972160306FA",
      "eventID": "ee3b4e0c-ab12-459b-9998-0a5a6f2e4015",
      "readOnly": false,
      "resources": [
        {
          "accountId": "222222222222",
          "type": "AWS::S3Outposts::Object",
          "ARN": "arn:aws:s3-outposts:us-east-1:YYY:outpost/op-01ac5d28a6a232904/
bucket/path/upload.sh"
        },
        {
          "accountId": "222222222222",
          "type": "AWS::S3Outposts::Bucket",
          "ARN": "arn:aws:s3-outposts:us-east-1:YYY:outpost/op-01ac5d28a6a232904/
bucket/"
```

```
        }
      ],
      "eventType": "AwsApiCall",
      "managementEvent": false,
      "recipientAccountId": "444455556666",
      "sharedEventID": "02759a4c-c040-4758-b84b-7cbaaf17747a",
      "edgeDeviceDetails": {
        "type": "outposts",
        "deviceId": "op-01ac5d28a6a232904"
      },
      "eventCategory": "Data"
    }
```

Mengaktifkan pencatatan CloudTrail peristiwa untuk bucket dan objek S3

Anda dapat menggunakan peristiwa CloudTrail data untuk mendapatkan informasi tentang bucket dan permintaan tingkat objek di Amazon S3. Untuk mengaktifkan peristiwa CloudTrail data untuk semua bucket Anda atau untuk daftar bucket tertentu, Anda harus membuat jejak secara manual. CloudTrail

> **Note**
>
> - Pengaturan default untuk CloudTrail adalah untuk menemukan hanya peristiwa manajemen. Periksa untuk memastikan bahwa Anda mengaktifkan peristiwa data untuk akun Anda.
> - Dengan bucket S3 yang menghasilkan beban kerja tinggi, Anda dapat dengan cepat menghasilkan ribuan log dalam waktu singkat. Perhatikan berapa lama Anda memilih untuk mengaktifkan peristiwa CloudTrail data untuk bucket yang sibuk.

CloudTrail menyimpan log peristiwa data Amazon S3 dalam ember S3 pilihan Anda. Pertimbangkan untuk menggunakan bucket secara terpisah Akun AWS untuk mengatur acara dengan lebih baik dari beberapa bucket yang mungkin Anda miliki ke tempat sentral untuk memudahkan kueri dan analisis. AWS Organizations membantu Anda membuat akun Akun AWS yang ditautkan ke akun yang memiliki bucket yang Anda pantau. Untuk informasi lebih lanjut, lihat Apa itu AWS Organizations? dalam AWS Organizations User Guide.

Saat Anda mencatat peristiwa data untuk jejak CloudTrail, Anda dapat memilih untuk menggunakan pemilih acara lanjutan atau pemilih acara dasar. Saat Anda membuat jejak di CloudTrail konsol

menggunakan pemilih peristiwa lanjutan, di bagian peristiwa data, Anda dapat memilih Log semua peristiwa untuk template pemilih Log untuk mencatat semua peristiwa tingkat objek. Saat Anda membuat jejak di CloudTrail konsol menggunakan pemilih acara dasar, di bagian peristiwa data, Anda dapat memilih kotak centang Pilih semua ember S3 di akun Anda untuk mencatat semua peristiwa tingkat objek.

> **Note**
>
> - Ini adalah praktik terbaik untuk membuat konfigurasi siklus hidup untuk bucket peristiwa AWS CloudTrail data Anda. Konfigurasikan konfigurasi siklus aktif untuk menghapus file log secara berkala setelah periode saat Anda yakin perlu mengauditnya. Hal ini mengurangi jumlah data yang dianalisis oleh Athena untuk setiap kueri. Untuk informasi selengkapnya, lihat Menyetel konfigurasi siklus hidup pada bucket.
> - Untuk informasi selengkapnya tentang format pencatatan, lihat Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail.
> - Untuk contoh cara menanyakan CloudTrail log, lihat posting Blog AWS Big Data Menganalisis Keamanan, Kepatuhan, dan Penggunaan Aktivitas Operasional AWS CloudTrail dan Amazon Athena.

Mengaktifkan pencatatan untuk objek dalam bucket menggunakan konsol tersebut

Anda dapat menggunakan konsol Amazon S3 untuk mengonfigurasi AWS CloudTrail jejak untuk mencatat peristiwa data untuk objek dalam bucket S3. CloudTrail mendukung pencatatan operasi API tingkat objek Amazon S3 sepertiGetObject,, dan. DeleteObject PutObject Peristiwa ini disebut peristiwa data.

Secara default, CloudTrail jejak tidak mencatat peristiwa data, tetapi Anda dapat mengonfigurasi jejak untuk mencatat peristiwa data untuk bucket S3 yang Anda tentukan, atau untuk mencatat peristiwa data untuk semua bucket Amazon S3 di bucket Anda. Akun AWS Untuk informasi selengkapnya, lihat Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail.

CloudTrail tidak mengisi peristiwa data dalam riwayat CloudTrail acara. Selain itu, tidak semua tindakan tingkat ember diisi dalam riwayat acara. CloudTrail Untuk informasi selengkapnya tentang tindakan API tingkat bucket Amazon S3 yang dilacak dengan logging, lihat. CloudTrail Tindakan tingkat ember Amazon S3 yang dilacak dengan logging CloudTrail Untuk informasi selengkapnya

tentang cara menanyakan CloudTrail log, lihat artikel Pusat AWS Pengetahuan tentang penggunaan pola filter CloudWatch Log Amazon dan Amazon Athena untuk menanyakan CloudTrail log.

Untuk mengonfigurasi jejak guna mencatat peristiwa data bagi bucket S3, Anda dapat menggunakan konsol AWS CloudTrail tersebut atau konsol Amazon S3. Jika Anda mengonfigurasi jejak untuk mencatat peristiwa data untuk semua bucket Amazon S3 di Akun AWS Anda, lebih mudah menggunakan konsol. CloudTrail Untuk informasi tentang menggunakan CloudTrail konsol untuk mengonfigurasi jejak untuk mencatat peristiwa data S3, lihat Peristiwa data di Panduan AWS CloudTrail Pengguna.

> **Important**
>
> Biaya tambahan berlaku untuk peristiwa data. Untuk informasi selengkapnya, lihat harga AWS CloudTrail.

Prosedur berikut menunjukkan cara menggunakan konsol Amazon S3 untuk mengonfigurasi CloudTrail jejak untuk mencatat peristiwa data untuk bucket S3.

Untuk mengaktifkan pencatatan peristiwa CloudTrail data untuk objek dalam bucket S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket.
3. Pilih Properti.
4. Di bawah peristiwa AWS CloudTrail data, pilih Konfigurasikan dalam CloudTrail.

   Anda dapat membuat CloudTrail jejak baru atau menggunakan kembali jejak yang ada dan mengonfigurasi peristiwa data Amazon S3 untuk dicatat di jejak Anda. Untuk informasi tentang cara membuat jejak di CloudTrail konsol, lihat Membuat dan memperbarui jejak dengan konsol di Panduan AWS CloudTrail Pengguna. Untuk informasi tentang cara mengonfigurasi pencatatan peristiwa data Amazon S3 di CloudTrail konsol, lihat Mencatat peristiwa data untuk Objek Amazon S3 di AWS CloudTrail Panduan Pengguna.

> **Note**
>
> Jika Anda menggunakan CloudTrail konsol atau konsol Amazon S3 untuk mengonfigurasi jejak untuk mencatat peristiwa data untuk bucket S3, konsol Amazon S3 menunjukkan bahwa pencatatan tingkat objek diaktifkan untuk bucket.

Untuk menonaktifkan pencatatan peristiwa CloudTrail data untuk objek dalam bucket S3

1. Masuk ke AWS Management Console dan buka CloudTrail konsol di https://console.aws.amazon.com/cloudtrail/.
2. Di panel navigasi kiri, pilih Jalur.
3. Pilih nama jejak yang Anda buat untuk peristiwa log untuk bucket Anda.
4. Di halaman detail untuk jejak Anda, pilih Hentikan pencatatan di sudut kanan atas.
5. Pada kotak dialog yang muncul, pilih Hentikan pencatatan.

Untuk informasi tentang mengaktifkan pencatatan tingkat objek saat Anda membuat bucket S3, lihat Membuat bucket.

Untuk informasi selengkapnya tentang CloudTrail logging dengan bucket S3, lihat topik berikut:

- Melihat properti untuk bucket S3
- Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail
- Bekerja dengan File CloudTrail Log di Panduan AWS CloudTrail Pengguna

Mengidentifikasi permintaan Amazon S3 menggunakan CloudTrail

Di Amazon S3, Anda dapat mengidentifikasi permintaan menggunakan log AWS CloudTrail peristiwa. AWS CloudTrail adalah cara yang lebih disukai untuk mengidentifikasi permintaan Amazon S3, tetapi jika Anda menggunakan log akses server Amazon S3, lihat. the section called “Mengidentifikasi permintaan S3”

Topik

- Mengidentifikasi permintaan yang dibuat ke Amazon S3 dalam log CloudTrail
- Mengidentifikasi permintaan Amazon S3 Signature Version 2 dengan menggunakan CloudTrail

- Mengidentifikasi akses ke objek S3 dengan menggunakan CloudTrail

Mengidentifikasi permintaan yang dibuat ke Amazon S3 dalam log CloudTrail

Setelah mengatur CloudTrail untuk mengirimkan acara ke bucket, Anda akan mulai melihat objek masuk ke bucket tujuan Anda di konsol Amazon S3. Ini diformat sebagai berikut:

```
s3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/Region/yyyy/mm/dd
```

Peristiwa yang dicatat oleh CloudTrail disimpan sebagai objek gzipped JSON terkompresi di bucket S3 Anda. Untuk menemukan permintaan secara efisien, Anda harus menggunakan layanan seperti Amazon Athena untuk mengindeks dan menanyakan log. CloudTrail

Untuk informasi selengkapnya tentang CloudTrail dan Athena, lihat Membuat tabel untuk AWS CloudTrail log di Athena menggunakan proyeksi partisi di Panduan Pengguna Amazon Athena.

Mengidentifikasi permintaan Amazon S3 Signature Version 2 dengan menggunakan CloudTrail

Anda dapat menggunakan log CloudTrail peristiwa untuk mengidentifikasi versi tanda tangan API mana yang digunakan untuk menandatangani permintaan di Amazon S3. Kemampuan ini penting karena dukungan untuk Signature Version 2 akan dinonaktifkan (dihentikan). Setelah itu, Amazon S3 tidak akan menerima lagi permintaan yang menggunakan Signature Version 2, dan semua permintaan harus menggunakan penandatanganan Signature Version 4.

Kami sangat menyarankan agar Anda menggunakannya CloudTrail untuk membantu menentukan apakah alur kerja Anda menggunakan penandatanganan Signature Version 2. Perbaiki dengan meningkatkan pustaka dan kode untuk menggunakan Signature Version 4 guna mencegah dampaknya pada bisnis Anda.

Untuk informasi selengkapnya, lihat Pengumuman: AWS CloudTrail untuk Amazon S3 menambahkan bidang baru untuk audit keamanan yang ditingkatkan. AWS re:Post

> **Note**
>
> CloudTrail peristiwa untuk Amazon S3 menyertakan versi tanda tangan dalam detail permintaan dengan nama kunci '. `additionalEventData` Untuk menemukan versi tanda tangan pada permintaan yang dibuat untuk objek di Amazon S3 sepertiGET,, dan DELETE

permintaanPUT, Anda harus mengaktifkan peristiwa CloudTrail data. (Fitur ini dimatikan secara default.)

AWS CloudTrail adalah metode yang disukai untuk mengidentifikasi permintaan Signature Version 2. Jika Anda menggunakan pencatatan akses server Amazon S3, lihat Mengidentifikasi permintaan Signature Version 2 menggunakan pencatatan akses Amazon S3.

Topik

- Contoh kueri Athena untuk mengidentifikasi permintaan Tanda Tangan versi 2 Amazon S3
- Mempartisikan data Signature Version 2

Contoh kueri Athena untuk mengidentifikasi permintaan Tanda Tangan versi 2 Amazon S3

Example — Pilih semua peristiwa Signature Version 2, dan hanya cetak **`EventTime`**, **`S3_Action`**, **`Request_Parameters`**, **`Region`**, **`SourceIP`**, dan **`UserAgent`**

Pada kueri Athena berikut, ganti *`s3_cloudtrail_events_db.cloudtrail_table`* dengan detail Athena Anda, dan menambahkan atau menghapus batasnya sesuai dengan kebutuhan.

```
SELECT EventTime, EventName as S3_Action, requestParameters as Request_Parameters,
 awsregion as AWS_Region, sourceipaddress as Source_IP, useragent as User_Agent
FROM s3_cloudtrail_events_db.cloudtrail_table
WHERE eventsource='s3.amazonaws.com'
AND json_extract_scalar(additionalEventData, '$.SignatureVersion')='SigV2'
LIMIT 10;
```

Example — Pilih semua peminta yang mengirimkan lalu lintas Signature Version 2

```
SELECT useridentity.arn, Count(requestid) as RequestCount
FROM s3_cloudtrail_events_db.cloudtrail_table
WHERE eventsource='s3.amazonaws.com'
    and json_extract_scalar(additionalEventData, '$.SignatureVersion')='SigV2'
Group by useridentity.arn
```

Mempartisikan data Signature Version 2

Jika Anda memiliki data dalam jumlah large untuk kueri, Anda dapat meredam biaya dan runtime Athena dengan membuat tabel partisi.

Untuk melakukannya, buat tabel baru dengan partisi sebagai berikut.

```
CREATE EXTERNAL TABLE s3_cloudtrail_events_db.cloudtrail_table_partitioned(
     eventversion STRING,
     userIdentity STRUCT<
         type:STRING,
         principalid:STRING,
         arn:STRING,
         accountid:STRING,
         invokedby:STRING,
         accesskeyid:STRING,
         userName:STRING,
      sessioncontext:STRUCT<
                 attributes:STRUCT<
                 mfaauthenticated:STRING,
                 creationdate:STRING>,
                 sessionIssuer:STRUCT<
                 type:STRING,
                 principalId:STRING,
                 arn:STRING,
                 accountId:STRING,
                 userName:STRING>
             >
          >,
     eventTime STRING,
     eventSource STRING,
     eventName STRING,
     awsRegion STRING,
     sourceIpAddress STRING,
     userAgent STRING,
     errorCode STRING,
     errorMessage STRING,
     requestParameters STRING,
     responseElements STRING,
     additionalEventData STRING,
     requestId STRING,
     eventId STRING,
     resources ARRAY<STRUCT<ARN:STRING,accountId: STRING,type:STRING>>,
```

```
        eventType STRING,
        apiVersion STRING,
        readOnly STRING,
        recipientAccountId STRING,
        serviceEventDetails STRING,
        sharedEventID STRING,
        vpcEndpointId STRING
    )
    PARTITIONED BY (region string, year string, month string, day string)
    ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.orc.OrcSerde'
    STORED AS INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
    OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
    LOCATION 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/';
```

Lalu, buat partisi secara terpisah. Anda tidak bisa mendapatkan hasil dari tanggal yang belum Anda buat.

```
ALTER TABLE s3_cloudtrail_events_db.cloudtrail_table_partitioned ADD
    PARTITION (region= 'us-east-1', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/us-east-1/2019/02/19/'
    PARTITION (region= 'us-west-1', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/us-west-1/2019/02/19/'
    PARTITION (region= 'us-west-2', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/us-west-2/2019/02/19/'
    PARTITION (region= 'ap-southeast-1', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/ap-southeast-1/2019/02/19/'
    PARTITION (region= 'ap-southeast-2', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/ap-southeast-2/2019/02/19/'
    PARTITION (region= 'ap-northeast-1', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/ap-northeast-1/2019/02/19/'
    PARTITION (region= 'eu-west-1', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/eu-west-1/2019/02/19/'
    PARTITION (region= 'sa-east-1', year= '2019', month= '02', day= '19') LOCATION
 's3://DOC-EXAMPLE-BUCKET1/AWSLogs/111122223333/CloudTrail/sa-east-1/2019/02/19/';
```

Kemudian, Anda dapat membuat permintaan berdasarkan partisi ini, dan Anda tidak perlu memuat seluruh bucket.

```
SELECT useridentity.arn,
```

```
Count(requestid) AS RequestCount
FROM s3_cloudtrail_events_db.cloudtrail_table_partitioned
WHERE eventsource='s3.amazonaws.com'
AND json_extract_scalar(additionalEventData, '$.SignatureVersion')='SigV2'
AND region='us-east-1'
AND year='2019'
AND month='02'
AND day='19'
Group by useridentity.arn
```

Mengidentifikasi akses ke objek S3 dengan menggunakan CloudTrail

Anda dapat menggunakan log AWS CloudTrail peristiwa untuk mengidentifikasi permintaan akses objek Amazon S3 untuk peristiwa data sepertiGetObject,, dan DeleteObjectPutObject, dan menemukan informasi tambahan tentang permintaan tersebut.

Contoh berikut menunjukkan cara mendapatkan semua permintaan PUT objek untuk Amazon S3 dari log AWS CloudTrail peristiwa.

Topik

- Contoh kueri Athena untuk mengidentifikasi permintaan akses objek Amazon S3

Contoh kueri Athena untuk mengidentifikasi permintaan akses objek Amazon S3

Pada kueri Athena berikut, ganti *s3_cloudtrail_events_db.cloudtrail_table* dengan detail Athena Anda, dan modifikasi rentang tanggal sesuai kebutuhan.

Example — Pilih semua peristiwa yang memiliki permintaan akses **PUT** objek, dan hanya cetak **EventTime**, **EventSource**, **SourceIP**, **UserAgent**, **BucketName**, **object**, dan **UserARN**

```
SELECT
  eventTime,
  eventName,
  eventSource,
  sourceIpAddress,
  userAgent,
  json_extract_scalar(requestParameters, '$.bucketName') as bucketName,
  json_extract_scalar(requestParameters, '$.key') as object,
  userIdentity.arn as userArn
FROM
  s3_cloudtrail_events_db.cloudtrail_table
```

```
WHERE
  eventName = 'PutObject'
  AND eventTime BETWEEN '2019-07-05T00:00:00Z' and '2019-07-06T00:00:00Z'
```

Example — Pilih semua peristiwa yang memiliki permintaan akses objek **GET**, kemudian hanya cetak untuk **EventTime**, **EventSource**, **SourceIP**, **UserAgent**, **BucketName**, **object**, dan **UserARN**

```
SELECT
  eventTime,
  eventName,
  eventSource,
  sourceIpAddress,
  userAgent,
  json_extract_scalar(requestParameters, '$.bucketName') as bucketName,
  json_extract_scalar(requestParameters, '$.key') as object,
  userIdentity.arn as userArn
FROM
  s3_cloudtrail_events_db.cloudtrail_table
WHERE
  eventName = 'GetObject'
  AND eventTime BETWEEN '2019-07-05T00:00:00Z' and '2019-07-06T00:00:00Z'
```

Example — Pilih semua peristiwa pemohon anonim ke bucket dalam periode tertentu dan hanya cetak **EventTime**, **EventName**, **EventSource**, **SourceIP**, **UserAgent**, **BucketName**, **UserARN**, dan **AccountID**

```
SELECT
  eventTime,
  eventName,
  eventSource,
  sourceIpAddress,
  userAgent,
  json_extract_scalar(requestParameters, '$.bucketName') as bucketName,
  userIdentity.arn as userArn,
  userIdentity.accountId
FROM
  s3_cloudtrail_events_db.cloudtrail_table
WHERE
  userIdentity.accountId = 'anonymous'
  AND eventTime BETWEEN '2019-07-05T00:00:00Z' and '2019-07-06T00:00:00Z'
```

Example — Identifikasi semua permintaan yang memerlukan ACL untuk otorisasi

Contoh kueri Amazon Athena berikut menunjukkan cara mengidentifikasi semua permintaan ke bucket S3 Anda yang memerlukan daftar kontrol akses (ACL) untuk otorisasi. Jika permintaan memerlukan ACL untuk otorisasi, `aclRequired` nilai dalam `additionalEventData` adalah `Yes`. Jika tidak ada ACL yang diperlukan, `aclRequired` tidak akan ada. Anda dapat menggunakan informasi ini untuk memigrasikan izin ACL tersebut ke kebijakan bucket yang sesuai. Setelah membuat kebijakan bucket ini, Anda dapat menonaktifkan ACL untuk bucket ini. Untuk informasi selengkapnya tentang menonaktifkan ACL, lihat Prasyarat untuk menonaktifkan ACL.

```
SELECT
  eventTime,
  eventName,
  eventSource,
  sourceIpAddress,
  userAgent,
  userIdentity.arn as userArn,
  json_extract_scalar(requestParameters, '$.bucketName') as bucketName,
  json_extract_scalar(requestParameters, '$.key') as object,
  json_extract_scalar(additionalEventData, '$.aclRequired') as aclRequired
FROM
  s3_cloudtrail_events_db.cloudtrail_table
WHERE
  json_extract_scalar(additionalEventData, '$.aclRequired') = 'Yes'
  AND eventTime BETWEEN '2022-05-10T00:00:00Z' and '2022-08-10T00:00:00Z'
```

Note

- Contoh kueri ini juga dapat berguna untuk pemantauan keamanan. Anda dapat meninjau hasil untuk panggilan `PutObject` atau `GetObject` dari alamat IP yang tidak terduga atau tidak sah atau pemohon, dan untuk mengidentifikasi permintaan anonim ke bucket Anda.
- Kueri ini hanya mengambil informasi dari waktu saat pencatatan log diaktifkan.

Jika Anda menggunakan log akses server Amazon S3, lihat Mengidentifikasi permintaan akses objek dengan menggunakan pencatatan akses Amazon S3.

Pencatatan permintaan dengan pencatatan akses server

Pencatatan akses server menyediakan catatan terperinci untuk permintaan yang dilakukan ke bucket. Log akses server bermanfaat untuk berbagai macam aplikasi. Misalnya, informasi log akses dapat berguna dalam audit keamanan dan akses. Informasi ini juga dapat membantu Anda untuk mempelajari basis pelanggan Anda, serta memahami tagihan Amazon S3.

> **Note**
>
> Log akses server tidak mencatat informasi tentang kesalahan pengalihan wilayah yang salah untuk Wilayah yang diluncurkan setelah 20 Maret 2019. Kesalahan pengalihan wilayah yang salah terjadi saat permintaan objek atau bucket dibuat di luar Wilayah tempat bucket berada.

Bagaimana cara mengaktifkan log pengiriman?

Untuk mengaktifkan log pengiriman, lakukan langkah-langkah basic berikut. Untuk rincian selengkapnya, lihat Mengaktifkan pencatatan akses server Amazon S3.

1. Berikan nama bucket tujuan (juga dikenal sebagai bucket target). Bucket ini adalah tempat Anda ingin Amazon S3 menyimpan pencatatan akses sebagai objek. Bucket sumber dan tujuan harus berada di Wilayah AWS yang sama, dan dimiliki oleh akun yang sama. Bucket tujuan tidak boleh memiliki konfigurasi periode retensi default Kunci Objek S3. Bucket tujuan juga harus tidak mengaktifkan Requester Pays.

   Anda dapat mengirimkan log ke bucket mana pun milik Anda yang berada di Wilayah yang sama dengan bucket sumber, termasuk bucket sumber itu sendiri. Tetapi untuk manajemen log yang lebih sederhana, kami sarankan agar Anda menyimpan log akses dalam bucket yang berbeda.

   Saat bucket sumber dan bucket tujuan Anda merupakan bucket yang sama, akan dibuat log tambahan untuk log yang ditulis ke bucket, yang akan membuat loop log tak terbatas. Kami tidak menyarankan melakukan hal ini karena dapat menyebabkan sedikit peningkatan dalam penagihan penyimpanan Anda. Selain itu, catatan tambahan tentang log mungkin akan menyulitkan Anda menemukan log yang Anda cari.

   Jika Anda memilih untuk menyimpan log akses di bucket sumber, sebaiknya Anda menentukan prefiks tujuan (juga dikenal sebagai prefiks target) untuk semua kunci objek log. Saat Anda menentukan prefiks, semua nama objek log dimulai dengan string umum, yang membuat objek log lebih mudah diidentifikasi.

2. (Opsional) Tetapkan prefiks tujuan ke semua kunci objek log Amazon S3. Awalan tujuan (juga dikenal sebagai prefiks target) menjadikan pencarian objek log lebih mudah. Misalnya, jika Anda menentukan nilai prefiks`logs/`, setiap objek log yang dibuat Amazon S3 dimulai dengan prefiks `logs/` di dalam kuncinya, misalnya:

   ```
   logs/2013-11-01-21-32-16-E568B2907131C0C0
   ```

   Jika Anda menentukan nilai prefiks `logs`, objek log muncul sebagai berikut:

   ```
   logs2013-11-01-21-32-16-E568B2907131C0C0
   ```

   Prefiks juga berguna untuk membedakan bucket sumber jika beberapa bucket mencatat ke bucket tujuan yang sama.

   Prefiks ini juga dapat membantu saat Anda menghapus pencatatan. Misalnya, Anda dapat menetapkan aturan konfigurasi siklus aktif bagi Amazon S3 untuk menghapus objek dengan prefiks kunci tertentu. Untuk informasi selengkapnya, lihat Menghapus file log Amazon S3.

3. (Opsional) Atur izin sehingga orang lain dapat mengakses log yang dihasilkan. Secara default, hanya pemilik bucket yang selalu memiliki akses penuh ke objek log. Jika bucket tujuan menggunakan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan S3 Object untuk menonaktifkan daftar kontrol akses (ACL), Anda tidak dapat memberikan izin dalam pemberian tujuan (juga dikenal sebagai pemberian target) yang menggunakan ACL. Namun, Anda dapat memperbarui kebijakan bucket untuk bucket tujuan untuk memberikan akses kepada orang lain. Untuk informasi selengkapnya, lihat Identity and Access Management untuk Amazon S3 dan Izin untuk pengiriman log.

4. (Opsional) Tetapkan format kunci objek log untuk file log. Anda memiliki dua opsi untuk format kunci objek log (juga dikenal sebagai format kunci objek target):

   - N on-date-based partisi - Ini adalah format kunci objek log asli. Jika Anda memilih format ini, format kunci file log muncul sebagai berikut:

     ```
     [DestinationPrefix][YYYY]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString]
     ```

     Misalnya, jika Anda menentukan `logs/` sebagai prefiks, objek log Anda diberi nama seperti ini:

     ```
     logs/2013-11-01-21-32-16-E568B2907131C0C0
     ```

- Partisi berbasis tanggal–Jika Anda memilih partisi berbasis tanggal, Anda dapat memilih waktu peristiwa atau waktu pengiriman untuk file log sebagai sumber tanggal yang digunakan dalam format log. Format ini membuatnya lebih mudah untuk mengkueri log.

  Jika Anda memilih partisi berbasis tanggal, format kunci file log akan muncul sebagai berikut:

  ```
  [DestinationPrefix][SourceAccountId]/[SourceRegion]/[SourceBucket]/[YYYY]/[MM]/
  [DD]/[YYYY]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString]
  ```

  Misalnya, jika Anda menentukan `logs/` sebagai prefiks target, objek log Anda diberi nama seperti ini:

  ```
  logs/123456789012/us-west-2/DOC-EXAMPLE-SOURCE-
  BUCKET/2023/03/01/2023-03-01-21-32-16-E568B2907131C0C0
  ```

  Untuk pengiriman waktu pengiriman, waktu dalam nama file log sesuai dengan waktu pengiriman untuk file log.

  Untuk pengiriman waktu peristiwa, tahun, bulan, dan hari sesuai dengan hari di mana peristiwa itu terjadi, dan jam, menit dan detik diatur ke `00` dalam kunci. Log yang dikirimkan dalam file log ini hanya untuk hari tertentu.

Jika Anda mengonfigurasi log melalui AWS Command Line Interface (AWS CLI), AWS SDK, atau Amazon S3 REST API, `TargetObjectKeyFormat` gunakan untuk menentukan format kunci objek log. Untuk menentukan non-date-based partisi, gunakan. `SimplePrefix` Untuk menentukan partisi berbasis data, gunakan `PartitionedPrefix`. Jika Anda menggunakan`PartitionedPrefix`, gunakan `PartitionDateSource` untuk menentukan antara `EventTime`, atau `DeliveryTime`.

Untuk `SimplePrefix`, format kunci file log muncul sebagai berikut:

```
[TargetPrefix][YYYY]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString]
```

Untuk `PartitionedPrefix` dengan waktu peristiwa atau waktu pengiriman, format kunci file log muncul sebagai berikut:

```
[TargetPrefix][SourceAccountId]/[SourceRegion]/[SourceBucket]/[YYYY]/[MM]/[DD]/
[YYYY]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString]
```

Format kunci objek log

Amazon S3 menggunakan format kunci objek berikut untuk objek log yang diunggahnya di bucket tujuan:

- N on-date-based partisi - Ini adalah format kunci objek log asli. Jika Anda memilih format ini, format kunci file log muncul sebagai berikut:

  ```
  [DestinationPrefix][YYYY]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString]
  ```

- Partisi berbasis tanggal–Jika Anda memilih partisi berbasis tanggal, Anda dapat memilih waktu peristiwa atau waktu pengiriman untuk file log sebagai sumber tanggal yang digunakan dalam format log. Format ini membuatnya lebih mudah untuk mengkueri log.

  Jika Anda memilih partisi berbasis tanggal, format kunci file log akan muncul sebagai berikut:

  ```
  [DestinationPrefix][SourceAccountId]/[SourceRegion]/[SourceBucket]/[YYYY]/[MM]/[DD]/
  [YYYY]-[MM]-[DD]-[hh]-[mm]-[ss]-[UniqueString]
  ```

Dalam kunci objek log, `YYYY`, `MM`, `DD`, `hh`, `mm`, dan `ss` merupakan digit tahun, bulan, hari, jam, menit, dan detik (masing-masing). Tanggal dan waktu ini berada dalam Waktu Universal Terkoordinasi (UTC).

Berkas log yang dikirimkan pada waktu tertentu dapat berisi catatan yang ditulis kapan pun sebelum waktu tersebut. Tidak ada cara untuk mengetahui apakah semua catatan log untuk interval waktu tertentu telah dikirim atau tidak.

Komponen `UniqueString` pada kunci ada untuk mencegah penimpaan file. Tidak memiliki makna, dan perangkat lunak pemroses log harus mengabaikannya.

Bagaimana log dikirimkan?

Amazon S3 secara berkala mengumpulkan catatan akses, mengonsolidasikan catatan dalam file log, kemudian mengunggah file log ke bucket tujuan sebagai objek log. Jika Anda mengaktifkan

pencatatan log ke beberapa bucket sumber yang mengidentifikasi bucket tujuan yang sama, bucket tujuan akan memiliki catatan akses untuk semua bucket sumber tersebut. Namun demikian, setiap objek catatan melaporkan arsip log akses untuk bucket sumber spesifik.

Amazon S3 menggunakan akun pengiriman log kustom untuk menulis log akses server. Penulisan ini tunduk pada pembatasan kontrol akses biasa. Kami menyarankan Anda memperbarui kebijakan bucket pada bucket tujuan untuk memberikan akses ke pengguna utama layanan pencatatan (`logging.s3.amazonaws.com`) kepada pengiriman log akses. Anda juga dapat memberikan akses kepada pengiriman log akses ke grup pengiriman log S3 melalui daftar kontrol akses (ACL) bucket Anda. Namun, pemberian akses ke grup pengiriman log S3 dengan menggunakan bucket ACL Anda tidak disarankan.

Saat mengaktifkan pencatatan akses server dan memberikan akses untuk pengiriman log akses melalui kebijakan bucket tujuan, Anda harus memperbarui kebijakan untuk mengizinkan akses `s3:PutObject` bagi pengguna utama layanan pencatatan. Jika Anda menggunakan konsol Amazon S3 untuk mengaktifkan pencatatan akses server, konsol akan secara otomatis memperbarui kebijakan bucket tujuan untuk memberikan izin ini kepada pengguna utama layanan logging. Untuk informasi selengkapnya tentang pemberian izin untuk pengiriman log akses server, lihat Izin untuk pengiriman log.

> **Note**
>
> S3 tidak mendukung pengiriman CloudTrail log atau log akses server ke pemohon atau pemilik bucket untuk permintaan titik akhir VPC saat kebijakan titik akhir VPC menyangkalnya atau untuk permintaan yang gagal sebelum kebijakan VPC dievaluasi.

Pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek S3

Jika bucket tujuan menggunakan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek, ACL akan dinonaktifkan dan tidak lagi memengaruhi izin. Anda harus memperbarui kebijakan bucket pada bucket tujuan untuk memberikan akses ke pengguna utama layanan logging. Untuk informasi selengkapnya tentang Kepemilikan Objek, lihat Berikan akses ke grup pengiriman log S3 untuk pencatatan akses server.

Pengiriman log server dengan upaya terbaik

Catatan log akses server disampaikan atas dasar upaya terbaik. Sebagian besar permintaan bucket yang dikonfigurasi dengan benar untuk mencatat hasil dalam catatan log yang dikirim. Sebagian besar catatan log dikirim dalam beberapa jam setelah log dicatat, tetapi dapat dikirimkan lebih sering.

Kelengkapan dan ketepatan waktu pencatatan server tidak dijamin. Catatan log untuk permintaan tertentu mungkin dikirim dalam waktu lama setelah permintaan diproses, atau mungkin tidak dikirimkan sama sekali. Ada kemungkinan bahwa Anda bahkan mungkin melihat duplikasi catatan log. Tujuan log server adalah untuk memberikan Anda ide tentang sifat lalu lintas terhadap bucket Anda. Meskipun catatan akuntasi log jarang hilang atau digandakan, perlu diketahui bahwa pencatatan akuntasi server tidak dimaksudkan untuk menjadi pencatatan akuntasi lengkap atas semua permintaan.

Karena sifat upaya terbarik dari pencatatan log, laporan penggunaan Anda dapat mencakup satu permintaan akses atau lebih yang tidak muncul di log server yang dikirim. Anda dapat menemukan laporan penggunaan ini di bawah Laporan biaya & penggunaan di konsol AWS Billing and Cost Management .

Perubahan status pencatatan log bucket memerlukan waktu

Perubahan status pencatatan log pada bucket memerlukan waktu untuk benar-benar memengaruhi pengiriman file log. Misalnya, jika Anda mengaktifkan pencatatan log untuk bucket, beberapa permintaan yang dilakukan di jam berikutnya mungkin akan dicatat, dan yang lainnya mungkin tidak. Misalkan Anda mengubah bucket tujuan untuk pencatatan log dari bucket A ke bucket B. Untuk jam berikutnya, beberapa catatan mungkin akan terus dikirimkan ke bucket A, sedangkan yang lain mungkin dikirimkan ke bucket tujuan B baru. Dalam semua kasus, pengaturan baru tersebut pada akhirnya akan berlaku tanpa tindakan lebih lanjut dari pihak Anda.

Untuk informasi lebih lanjut tentang pencatatan dan file log, lihat bagian berikut:

Topik

- Mengaktifkan pencatatan akses server Amazon S3
- Server Amazon S3 mengakses format log
- Menghapus file log Amazon S3
- Menggunakan log akses server Amazon S3 untuk mengidentifikasi permintaan

Mengaktifkan pencatatan akses server Amazon S3

Pencatatan akses server menyediakan catatan terperinci untuk permintaan yang dilakukan ke bucket Amazon S3 Anda. Log akses server bermanfaat untuk berbagai macam aplikasi. Misalnya, informasi log akses dapat berguna dalam audit keamanan dan akses. Informasi ini juga dapat membantu Anda untuk mempelajari basis pelanggan Anda, serta memahami tagihan Amazon S3.

Secara default, Amazon S3 tidak mengumpulkan log akses server. Saat Anda mengaktifkan pencatatan log, Amazon S3 mengirimkan pencatatan akses untuk bucket sumber ke bucket tujuan (juga dikenal sebagai bucket target) yang Anda pilih. Bucket tujuan harus sama Wilayah AWS dan Akun AWS sebagai bucket sumber.

Catatan log akses berisi detail tentang permintaan yang dilakukan ke bucket. Informasi ini dapat mencakup jenis permintaan, sumber daya yang ditentukan dalam permintaan, dan waktu serta tanggal pemrosesan permintaan. Untuk informasi lebih lanjut tentang dasar pencatatan log, lihat Pencatatan permintaan dengan pencatatan akses server.

Important

- Tidak ada biaya tambahan untuk mengaktifkan pencatatan akses server di bucket Amazon S3. Namun, semua file log yang dikirim sistem kepada Anda akan dikenakan biaya penyimpanan yang biasa. (Anda dapat menghapus file log ini kapan saja.) Kami tidak menilai biaya transfer data untuk pengiriman file log, tetapi kami memang mengenakan tarif transfer data normal untuk mengakses file log.
- bucket tujuan Anda seharusnya tidak memiliki akses server pencatatan yang diaktifkan. Anda dapat mengirimkan log ke bucket mana pun milik Anda yang berada di Wilayah yang sama dengan bucket sumber, termasuk bucket sumber itu sendiri. Namun, mengirimkan log ke bucket sumber akan menyebabkan loop log yang tak terbatas dan tidak disarankan. Tetapi untuk manajemen log yang lebih sederhana, kami sarankan agar Anda menyimpan log akses dalam bucket yang berbeda. Untuk informasi selengkapnya, lihat Bagaimana cara mengaktifkan log pengiriman?
- Bucket S3 yang mengaktifkan Kunci Objek S3 tidak dapat digunakan sebagai bucket tujuan untuk pencatatan akses server. Bucket tujuan Anda tidak boleh memiliki konfigurasi periode retensi default.
- Bucket tujuan tidak boleh mengaktifkan Pembayaran oleh Pemohon.
- Anda dapat menggunakan enkripsi bucket default pada bucket tujuan hanya jika Anda menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3), yang

menggunakan Standar Enkripsi Lanjutan 256 bit (AES-256). Enkripsi sisi server default dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS) tidak didukung.

Anda dapat mengaktifkan atau nonaktifkan pencatatan log akses server menggunakan konsol Amazon S3, API Amazon S3, AWS Command Line Interface (AWS CLI), atau SDK AWS .

Izin untuk pengiriman log

Amazon S3 menggunakan akun pengiriman log kustom untuk menulis log akses server. Penulisan ini tunduk pada pembatasan kontrol akses biasa. Untuk pengiriman log akses, Anda harus memberikan akses kepada pengguna utama layanan pencatatan (`logging.s3.amazonaws.com`) ke bucket tujuan Anda.

Untuk memberikan izin ke Amazon S3 untuk pengiriman log, Anda dapat menggunakan kebijakan bucket atau daftar kontrol akses bucket (ACL), tergantung pada pengaturan Kepemilikan S3 Object bucket tujuan. Namun, kami menyarankan Anda menggunakan kebijakan bucket, bukan ACL.

Pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek S3

Jika bucket tujuan menggunakan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek, ACL akan dinonaktifkan dan tidak lagi memengaruhi izin. Dalam hal ini, Anda harus memperbarui kebijakan bucket untuk bucket tujuan untuk memberikan akses ke pengguna utama layanan logging. Anda tidak dapat memperbarui ACL bucket untuk memberikan akses ke grup pengiriman log S3. Anda juga tidak dapat menyertakan pemberian tujuan (juga dikenal sebagai pemberian target) dalam konfigurasi PutBucketLogging Anda.

Untuk informasi tentang memigrasi ACL bucket yang ada untuk pengiriman log akses ke kebijakan bucket, lihat Berikan akses ke grup pengiriman log S3 untuk pencatatan akses server. Untuk informasi selengkapnya tentang Kepemilikan Objek, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda. Saat Anda membuat bucket baru, ACL dinonaktifkan secara default.

memberikan akses dengan menggunakan kebijakan bucket

Untuk memberikan akses dengan menggunakan kebijakan bucket di bucket tujuan, perbarui kebijakan bucket untuk memberikan izin `s3:PutObject` kepada pengguna utama layanan logging. Jika Anda menggunakan konsol Amazon S3 untuk mengaktifkan pencatatan akses server, konsol secara otomatis memperbarui kebijakan bucket pada bucket tujuan untuk memberikan izin ini kepada pengguna utama layanan pencatatan. Jika Anda mengaktifkan pencatatan akses server secara

terprogram, Anda harus memperbarui kebijakan bucket secara manual untuk bucket tujuan untuk memberikan akses ke pengguna utama layanan logging.

Untuk contoh kebijakan bucket yang memberikan akses ke pengguna utama layanan logging, lihat the section called "Berikan izin kepada pengguna utama layanan logging dengan menggunakan kebijakan bucket".

memberikan akses dengan menggunakan bucket ACL

Anda dapat menggunakan ACL bucket secara bergantian untuk memberikan akses pengiriman log akses. Anda menambahkan entri pemberian ke ACL bucket yang memberikan izin `WRITE` dan `READ_ACP` ke grup pengiriman log S3. Namun, pemberian akses ke grup pengiriman log S3 dengan menggunakan bucket ACL tidak disarankan. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda. Untuk informasi tentang memigrasi ACL bucket yang ada untuk pengiriman log akses ke kebijakan bucket, lihat Berikan akses ke grup pengiriman log S3 untuk pencatatan akses server. Untuk contoh ACL yang memberikan akses ke pengguna utama layanan logging, lihat the section called "Berikan izin ke grup pengiriman log dengan menggunakan bucket ACL".

Berikan izin kepada pengguna utama layanan logging dengan menggunakan kebijakan bucket

Contoh kebijakan bucket ini memberikan `s3:PutObject` izin kepada pengguna utama layanan logging (`logging.s3.amazonaws.com`). Untuk menggunakan kebijakan bucket ini, ganti *`user input placeholders`* dengan informasi Anda sendiri. Dalam kebijakan berikut, *`example-s3-destination-bucket`* adalah bucket tujuan tempat log akses server akan dikirimkan, dan *`example-s3-source-bucket`* merupakan bucket sumber. *`EXAMPLE-LOGGING-PREFIX`*adalah awalan tujuan opsional (juga dikenal sebagai awalan target) yang ingin Anda gunakan untuk objek log Anda. *`SOURCE-ACCOUNT-ID`*adalah Akun AWS yang memiliki ember sumber.

> **Note**
>
> Jika terdapat pernyataan `Deny` dalam kebijakan bucket Anda, pastikan pernyataan tersebut tidak mencegah Amazon S3 mengirimkan pencatatan akses.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ServerAccessLogsPolicy",
```

```
            "Effect": "Allow",
            "Principal": {
                "Service": "logging.s3.amazonaws.com"
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::example-s3-destination-bucket/EXAMPLE-LOGGING-
PREFIX*",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:::example-s3-source-bucket"
                },
                "StringEquals": {
                    "aws:SourceAccount": "SOURCE-ACCOUNT-ID"
                }
            }
        }
    ]
}
```

Berikan izin ke grup pengiriman log dengan menggunakan bucket ACL

> **Note**
>
> Sebagai praktik keamanan terbaik, Amazon S3 menonaktifkan daftar kontrol akses (ACL) secara default di semua bucket baru. Untuk informasi selengkapnya tentang izin ACL di konsol Amazon S3, lihat Mengonfigurasi ACL.

Meskipun kami tidak merekomendasikan pendekatan ini, Anda dapat memberikan izin ke grup pengiriman log dengan menggunakan bucket ACL. Namun, jika bucket tujuan menggunakan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek, Anda tidak dapat menyetel ACL bucket atau objek. Anda juga tidak dapat menyertakan pemberian tujuan (juga dikenal sebagai pemberian target) dalam konfigurasi PutBucketLogging Anda. Sebagai gantinya, Anda harus menggunakan kebijakan bucket untuk memberikan akses ke pengguna utama layanan logging (`logging.s3.amazonaws.com`). Untuk informasi selengkapnya, lihat Izin untuk pengiriman log.

Di bucket ACL, grup pengiriman log direpresentasikan dengan URL berikut:

```
http://acs.amazonaws.com/groups/s3/LogDelivery
```

Untuk memberikan izin `WRITE` dan `READ_ACP` (ACL baca), menambahkan pemberian berikut ini ke ACL bucket tujuan:

```
<Grant>
    <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/s3/LogDelivery</URI>
    </Grantee>
    <Permission>WRITE</Permission>
</Grant>
<Grant>
    <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/s3/LogDelivery</URI>
    </Grantee>
    <Permission>READ_ACP</Permission>
</Grant>
```

Untuk contoh penambahan pemberian ACL secara terprogram, lihat Mengonfigurasi ACL.

> **Important**
>
> Saat Anda mengaktifkan pencatatan akses server Amazon S3 dengan menggunakan AWS CloudFormation pada bucket dan Anda menggunakan ACL untuk memberikan akses ke grup pengiriman log S3, Anda juga harus menambahkan `"AccessControl": "LogDeliveryWrite"`ke template Anda. CloudFormation Melakukannya penting karena Anda dapat memberikan izin tersebut hanya dengan membuat ACL untuk bucket, tetapi Anda tidak dapat membuat ACL khusus untuk bucket. CloudFormation Anda hanya dapat menggunakan ACL kalengan dengan CloudFormation.

Untuk mengaktifkan pencatatan log akses server

Untuk mengaktifkan pencatatan akses server menggunakan konsol Amazon S3, Amazon S3 REST API AWS , SDK, AWS CLI dan, gunakan prosedur berikut.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di dalam daftar Bucket, pilih nama bucket yang ingin Anda aktifkan pencatatan akses server.
3. Pilih Properti.

4. Di Pencatatan akses server bagian, pilih Edit.
5. Di bawah Pencatatan akses server, pilih Aktifkan.
6. Di Bucket tujuan, tentukan bucket dan prefiks opsional. Jika Anda menentukan prefiks, sebaiknya sertakan garis miring ke depan (`/`) setelah prefiks agar lebih mudah untuk menemukan log Anda.

> **Note**
>
> Menentukan prefiks dengan garis miring (`/`) membuatnya lebih mudah bagi Anda untuk menemukan objek log. Misalnya, jika Anda menentukan nilai prefiks`logs/`, setiap objek log yang dibuat Amazon S3 dimulai dengan prefiks `logs/` pada kuncinya, sebagai berikut:
>
> ```
> logs/2013-11-01-21-32-16-E568B2907131C0C0
> ```
>
> Jika Anda menentukan nilai prefiks `logs`, objek log muncul sebagai berikut:
>
> ```
> logs2013-11-01-21-32-16-E568B2907131C0C0
> ```

7. Di bawah Format kunci objek log , lakukan salah satu langkah berikut:
   - Untuk memilih non-date-based partisi, pilih [DestinationPrefix] [YYYY] - [MM] - [DD] - [hh] - [mm] - [ss] - []. UniqueString
   - Untuk memilih partisi berbasis tanggal, pilih [DestinationPrefix] []/[]/[SourceAccountId]/[SourceRegionYYYYSourceBucket]/[MM]/[DD]/[YYYY] - [MM] - [DD] - [hh] - [mm] - [ss] - [], lalu pilih waktu acara S3 atau Waktu pengiriman file log. UniqueString
8. Pilih Simpan perubahan.

   Saat Anda mengaktifkan pencatatan log akses server pada bucket, konsol keduanya memungkinkan pencatatan log pada bucket sumber dan memperbarui kebijakan bucket untuk bucket tujuan guna memberikan izin `s3:PutObject` kepada pengguna utama layanan pencatatan log (`logging.s3.amazonaws.com`). Untuk informasi lebih lanjut tentang kebijakan bucket ini, lihat Berikan izin kepada pengguna utama layanan logging dengan menggunakan kebijakan bucket.

   Anda dapat melihat pencatatan di dalam bucket tujuan. Setelah Anda mengaktifkan pencatatan akses server, mungkin perlu beberapa jam sebelum pencatatan dikirim ke bucket target.

Untuk informasi selengkapnya tentang bagaimana dan kapan log dikirim, lihat Bagaimana log dikirimkan?

Untuk informasi selengkapnya, lihat Melihat properti untuk bucket S3.

Penggunaan API REST

Untuk mengaktifkan pencatatan log, Anda mengirimkan permintaan PutBucketLogging untuk menambahkan konfigurasi pencatatan log di bucket sumber. Permintaan menentukan bucket tujuan (juga dikenal sebagai bucket target) dan, secara opsional, prefiks yang akan digunakan dengan semua kunci objek log.

Contoh berikut mengidentifikasi *example-s3-destination-bucket* sebagai bucket tujuan, dan *logs/*sebagai prefiks.

```
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01">
  <LoggingEnabled>
    <TargetBucket>example-s3-destination-bucket</TargetBucket>
    <TargetPrefix>logs/</TargetPrefix>
  </LoggingEnabled>
</BucketLoggingStatus>
```

Contoh berikut mengidentifikasi *example-s3-destination-bucket* sebagai bucket tujuan, *logs/* sebagai prefiks, dan `EventTime` sebagai format kunci objek log.

```
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01">
  <LoggingEnabled>
    <TargetBucket>example-s3-destination-bucket</TargetBucket>
    <TargetPrefix>logs/</TargetPrefix>
    <TargetObjectKeyFormat>
      <PartitionedPrefix>
         <PartitionDateSource>EventTime</PartitionDateSource>
      </PartitionedPrefix>
  </TargetObjectKeyFormat>
  </LoggingEnabled>
</BucketLoggingStatus>
```

Objek log ditulis dan dimiliki oleh akun pengiriman log S3, dan pemilik bucket diberi izin penuh pada objek log. Anda dapat secara opsional menggunakan pemberian tujuan (juga dikenal sebagai pemberian target) untuk memberikan izin kepada pengguna lain sehingga mereka dapat mengakses log. Untuk informasi selengkapnya, lihat PutBucketLogging.

> **Note**
>
> Jika bucket tujuan menggunakan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek, Anda tidak dapat menggunakan pemberian tujuan untuk memberikan izin kepada pengguna lain. Untuk memberikan izin kepada orang lain, Anda dapat memperbarui kebijakan bucket di bucket tujuan. Untuk informasi selengkapnya, lihat Izin untuk pengiriman log.

Untuk mengambil konfigurasi logging pada bucket, gunakan operasi API GetBucketLogging.

Untuk menghapus konfigurasi pencatatan log, Anda mengirimkan `PutBucketLogging` permintaan dengan `BucketLoggingStatus` yang kosong:

```
<BucketLoggingStatus xmlns="http://doc.s3.amazonaws.com/2006-03-01">
</BucketLoggingStatus>
```

Untuk mengaktifkan logging pada bucket, Anda dapat menggunakan Amazon S3 API atau pustaka pembungkus AWS SDK.

Menggunakan AWS SDK

Contoh berikut mengaktifkan pencatatan log ke bucket. Anda harus membuat dua bucket, satu bucket sumber, dan satu bucket tujuan (target). Contoh ini memperbarui bucket ACL di bucket tujuan terlebih dahulu. Mereka kemudian memberikan ke grup pengiriman log izin yang diperlukan untuk menulis log ke bucket tujuan, dan kemudian mereka mengaktifkan pencatatan log ke bucket sumber.

Contoh ini tidak akan berfungsi pada bucket tujuan yang menggunakan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek.

Jika bucket tujuan (target) menggunakan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek, Anda tidak dapat mengatur ACL bucket atau objek. Anda juga tidak dapat menyertakan hibah tujuan (target) dalam PutBucketLoggingkonfigurasi Anda. Anda harus menggunakan kebijakan bucket untuk memberikan akses ke pengguna utama layanan pencatatan (`logging.s3.amazonaws.com`). Untuk informasi selengkapnya, lihat Izin untuk pengiriman log.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
using System;
using System.IO;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;
using Microsoft.Extensions.Configuration;

/// <summary>
/// This example shows how to enable logging on an Amazon Simple Storage
/// Service (Amazon S3) bucket. You need to have two Amazon S3 buckets for
/// this example. The first is the bucket for which you wish to enable
/// logging, and the second is the location where you want to store the
/// logs.
/// </summary>
public class ServerAccessLogging
{
    private static IConfiguration _configuration = null!;

    public static async Task Main()
    {
        LoadConfig();

        string bucketName = _configuration["BucketName"];
        string logBucketName = _configuration["LogBucketName"];
        string logObjectKeyPrefix = _configuration["LogObjectKeyPrefix"];
        string accountId = _configuration["AccountId"];

        // If the AWS Region defined for your default user is different
        // from the Region where your Amazon S3 bucket is located,
        // pass the Region name to the Amazon S3 client object's constructor.
        // For example: RegionEndpoint.USWest2 or RegionEndpoint.USEast2.
        IAmazonS3 client = new AmazonS3Client();
```

```
            try
            {
                // Update bucket policy for target bucket to allow delivery of
logs to it.
                await SetBucketPolicyToAllowLogDelivery(
                    client,
                    bucketName,
                    logBucketName,
                    logObjectKeyPrefix,
                    accountId);

                // Enable logging on the source bucket.
                await EnableLoggingAsync(
                    client,
                    bucketName,
                    logBucketName,
                    logObjectKeyPrefix);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine($"Error: {e.Message}");
            }
        }

        /// <summary>
        /// This method grants appropriate permissions for logging to the
        /// Amazon S3 bucket where the logs will be stored.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client which will be
used
        /// to apply the bucket policy.</param>
        /// <param name="sourceBucketName">The name of the source bucket.</param>
        /// <param name="logBucketName">The name of the bucket where logging
        /// information will be stored.</param>
        /// <param name="logPrefix">The logging prefix where the logs should be
delivered.</param>
        /// <param name="accountId">The account id of the account where the
source bucket exists.</param>
        /// <returns>Async task.</returns>
        public static async Task SetBucketPolicyToAllowLogDelivery(
            IAmazonS3 client,
            string sourceBucketName,
            string logBucketName,
```

```
            string logPrefix,
            string accountId)
        {
            var resourceArn = @"""arn:aws:s3:::" + logBucketName + "/" +
logPrefix + @"*""";

            var newPolicy = @"{
                                ""Statement"":[{
                                ""Sid"": ""S3ServerAccessLogsPolicy"",
                                ""Effect"": ""Allow"",
                                ""Principal"": { ""Service"":
""logging.s3.amazonaws.com"" },
                                ""Action"": [""s3:PutObject""],
                                ""Resource"": [" + resourceArn + @"],
                                ""Condition"": {
                                ""ArnLike"": { ""aws:SourceArn"":
""arn:aws:s3:::" + sourceBucketName + @""" },
                                ""StringEquals"": { ""aws:SourceAccount"": """ +
accountId + @""" }
                                        }
                                    }]
                                }";
            Console.WriteLine($"The policy to apply to bucket {logBucketName} to
enable logging:");
            Console.WriteLine(newPolicy);

            PutBucketPolicyRequest putRequest = new PutBucketPolicyRequest
            {
                BucketName = logBucketName,
                Policy = newPolicy,
            };
            await client.PutBucketPolicyAsync(putRequest);
            Console.WriteLine("Policy applied.");
        }

        /// <summary>
        /// This method enables logging for an Amazon S3 bucket. Logs will be
stored
        /// in the bucket you selected for logging. Selected prefix
        /// will be prepended to each log object.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client which will be
used
```

```
        /// to configure and apply logging to the selected Amazon S3 bucket.</
param>
        /// <param name="bucketName">The name of the Amazon S3 bucket for which
 you
        /// wish to enable logging.</param>
        /// <param name="logBucketName">The name of the Amazon S3 bucket where
 logging
        /// information will be stored.</param>
        /// <param name="logObjectKeyPrefix">The prefix to prepend to each
        /// object key.</param>
        /// <returns>Async task.</returns>
        public static async Task EnableLoggingAsync(
            IAmazonS3 client,
            string bucketName,
            string logBucketName,
            string logObjectKeyPrefix)
        {
            Console.WriteLine($"Enabling logging for bucket {bucketName}.");
            var loggingConfig = new S3BucketLoggingConfig
            {
                TargetBucketName = logBucketName,
                TargetPrefix = logObjectKeyPrefix,
            };

            var putBucketLoggingRequest = new PutBucketLoggingRequest
            {
                BucketName = bucketName,
                LoggingConfig = loggingConfig,
            };
            await client.PutBucketLoggingAsync(putBucketLoggingRequest);
            Console.WriteLine($"Logging enabled.");
        }

        /// <summary>
        /// Loads configuration from settings files.
        /// </summary>
        public static void LoadConfig()
        {
            _configuration = new ConfigurationBuilder()
                .SetBasePath(Directory.GetCurrentDirectory())
                .AddJsonFile("settings.json") // Load settings from .json file.
                .AddJsonFile("settings.local.json", true) // Optionally, load
 local settings.
                .Build();
```

```
        }
    }
```

- Untuk detail API, lihat PutBucketLoggingdi Referensi AWS SDK for .NET API.

Java

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.BucketLoggingStatus;
import software.amazon.awssdk.services.s3.model.LoggingEnabled;
import software.amazon.awssdk.services.s3.model.PartitionedPrefix;
import software.amazon.awssdk.services.s3.model.PutBucketLoggingRequest;
import software.amazon.awssdk.services.s3.model.TargetObjectKeyFormat;

// Class to set a bucket policy on a target S3 bucket and enable server access
 logging on a source S3 bucket.
public class ServerAccessLogging {
    private static S3Client s3Client;

    public static void main(String[] args) {
        String sourceBucketName = "SOURCE-BUCKET";
        String targetBucketName = "TARGET-BUCKET";
        String sourceAccountId = "123456789012";
        String targetPrefix = "logs/";

        // Create S3 Client.
        s3Client = S3Client.builder().
                region(Region.US_EAST_2)
                .build();

        // Set a bucket policy on the target S3 bucket to enable server access
 logging by granting the
        // logging.s3.amazonaws.com principal permission to use the PutObject
 operation.
        ServerAccessLogging serverAccessLogging = new ServerAccessLogging();
        serverAccessLogging.setTargetBucketPolicy(sourceAccountId, sourceBucketName,
 targetBucketName);

        // Enable server access logging on the source S3 bucket.
```

```
        serverAccessLogging.enableServerAccessLogging(sourceBucketName,
 targetBucketName,
                targetPrefix);

    }

    // Function to set a bucket policy on the target S3 bucket to enable server
 access logging by granting the
    // logging.s3.amazonaws.com principal permission to use the PutObject operation.
    public void setTargetBucketPolicy(String sourceAccountId, String
 sourceBucketName, String targetBucketName) {
        String policy = "{\n" +
                "    \"Version\": \"2012-10-17\",\n" +
                "    \"Statement\": [\n" +
                "        {\n" +
                "            \"Sid\": \"S3ServerAccessLogsPolicy\",\n" +
                "            \"Effect\": \"Allow\",\n" +
                "            \"Principal\": {\"Service\": \"logging.s3.amazonaws.com
\"},\n" +
                "            \"Action\": [\n" +
                "                \"s3:PutObject\"\n" +
                "            ],\n" +
                "            \"Resource\": \"arn:aws:s3:::" + targetBucketName + "/*
\",\n" +
                "            \"Condition\": {\n" +
                "                \"ArnLike\": {\n" +
                "                    \"aws:SourceArn\": \"arn:aws:s3:::" +
 sourceBucketName + "\"\n" +
                "                },\n" +
                "                \"StringEquals\": {\n" +
                "                    \"aws:SourceAccount\": \"" + sourceAccountId +
 "\"\n" +
                "                }\n" +
                "            }\n" +
                "        }\n" +
                "    ]\n" +
                "}";
        s3Client.putBucketPolicy(b -> b.bucket(targetBucketName).policy(policy));
    }

    // Function to enable server access logging on the source S3 bucket.
    public void enableServerAccessLogging(String sourceBucketName, String
 targetBucketName,
            String targetPrefix) {
```

```
        TargetObjectKeyFormat targetObjectKeyFormat =
TargetObjectKeyFormat.builder()

.partitionedPrefix(PartitionedPrefix.builder().partitionDateSource("EventTime").build())
                .build();
        LoggingEnabled loggingEnabled = LoggingEnabled.builder()
                .targetBucket(targetBucketName)
                .targetPrefix(targetPrefix)
                .targetObjectKeyFormat(targetObjectKeyFormat)
                .build();
        BucketLoggingStatus bucketLoggingStatus = BucketLoggingStatus.builder()
                .loggingEnabled(loggingEnabled)
                .build();
        s3Client.putBucketLogging(PutBucketLoggingRequest.builder()
                .bucket(sourceBucketName)
                .bucketLoggingStatus(bucketLoggingStatus)
                .build());
    }

}
```

Menggunakan AWS CLI

Kami menyarankan Anda membuat bucket logging khusus di setiap bucket S3 Wilayah AWS yang Anda miliki. Kemudian, mintalah log akses Amazon S3 dikirimkan ke bucket S3 tersebut. Untuk informasi dan contoh selengkapnya, lihat put-bucket-logging di Referensi AWS CLI .

Jika bucket tujuan (target) menggunakan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek, Anda tidak dapat mengatur ACL bucket atau objek. Anda juga tidak dapat menyertakan hibah tujuan (target) dalam PutBucketLoggingkonfigurasi Anda. Anda harus menggunakan kebijakan bucket untuk memberikan akses ke pengguna utama layanan pencatatan (`logging.s3.amazonaws.com`). Untuk informasi selengkapnya, lihat Izin untuk pengiriman log.

Example — Mengaktifkan pencatatan akses dengan lima bucket di dua Wilayah

Dalam contoh ini, Anda memiliki lima bucket berikut:

- `1-DOC-EXAMPLE-BUCKET1-us-east-1`
- `2-DOC-EXAMPLE-BUCKET1-us-east-1`
- `3-DOC-EXAMPLE-BUCKET1-us-east-1`

- `1-DOC-EXAMPLE-BUCKET1-us-west-2`
- `2-DOC-EXAMPLE-BUCKET1-us-west-2`

> **Note**
>
> Langkah terakhir dari prosedur berikut memberikan contoh skrip bash yang dapat Anda gunakan untuk membuat bucket logging Anda dan mengaktifkan log akses server pada bucket ini. Untuk menggunakan skrip tersebut, Anda harus membuat file `policy.json` dan `logging.json`, seperti yang dijelaskan dalam prosedur berikut.

1. Buat dua bucket pencatatan di Wilayah AS Barat (Oregon) dan AS Timur (Virginia Utara), dan beri mereka nama-nama berikut ini:
   - `DOC-EXAMPLE-BUCKET1-logs-us-east-1`
   - `DOC-EXAMPLE-BUCKET1-logs-us-west-2`
2. Kemudian dalam langkah-langkah ini, Anda akan mengaktifkan pencatatan akses server sebagai berikut:
   - `1-DOC-EXAMPLE-BUCKET1-us-east-1` menyimpan log ke bucket S3 `DOC-EXAMPLE-BUCKET1-logs-us-east-1` dengan prefiks `1-DOC-EXAMPLE-BUCKET1-us-east-1`
   - `2-DOC-EXAMPLE-BUCKET1-us-east-1` menyimpan log ke bucket S3 `DOC-EXAMPLE-BUCKET1-logs-us-east-1` dengan prefiks `2-DOC-EXAMPLE-BUCKET1-us-east-1`
   - `3-DOC-EXAMPLE-BUCKET1-us-east-1` menyimpan log ke bucket S3 `DOC-EXAMPLE-BUCKET1-logs-us-east-1` dengan prefiks `3-DOC-EXAMPLE-BUCKET1-us-east-1`
   - `1-DOC-EXAMPLE-BUCKET1-us-west-2` menyimpan log ke bucket S3 `DOC-EXAMPLE-BUCKET1-logs-us-west-2` dengan prefiks `1-DOC-EXAMPLE-BUCKET1-us-west-2`
   - `2-DOC-EXAMPLE-BUCKET1-us-west-2` menyimpan log ke bucket S3 `DOC-EXAMPLE-BUCKET1-logs-us-west-2` dengan prefiks `2-DOC-EXAMPLE-BUCKET1-us-west-2`
3. Untuk setiap bucket logging tujuan, berikan izin untuk pengiriman log akses server dengan menggunakan bucket ACL atau kebijakan bucket:
   - Perbarui kebijakan bucket (Disarankan)–Untuk memberikan izin kepada pengguna utama layanan logging, gunakan perintah `put-bucket-policy` berikut. Ganti *`example-s3-destination-bucket-logs`* dengan nama bucket tujuan Anda.

```
aws s3api put-bucket-policy --bucket example-s3-destination-bucket-logs --policy
 file://policy.json
```

`Policy.json` adalah dokumen JSON di dalam folder saat ini, yang berisi kebijakan bucket berikut. Untuk menggunakan kebijakan bucket ini, ganti *`user input placeholders`* dengan informasi Anda sendiri. Dalam kebijakan berikut, *`example-s3-destination-bucket-logs`*adalah bucket tujuan tempat log akses server akan dikirimkan, dan *`example-s3-source-bucket`* merupakan bucket sumber. *`SOURCE-ACCOUNT-ID`* adalah Akun AWS yang memiliki bucket sumber.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ServerAccessLogsPolicy",
            "Effect": "Allow",
            "Principal": {
                "Service": "logging.s3.amazonaws.com"
            },
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::example-s3-destination-bucket-logs/*",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:::example-s3-source-bucket"
                },
                "StringEquals": {
                    "aws:SourceAccount": "SOURCE-ACCOUNT-ID"
                }
            }
        }
    ]
}
```

- Perbarui bucket ACL—Untuk memberikan izin ke grup pengiriman log S3, gunakan perintah `put-bucket-acl` berikut. Ganti *`example-s3-destination-bucket-logs`* dengan nama bucket tujuan (target) Anda.

```
aws s3api put-bucket-acl --bucket example-s3-destination-bucket-logs  --grant-
write URI=http://acs.amazonaws.com/groups/s3/LogDelivery --grant-read-acp
 URI=http://acs.amazonaws.com/groups/s3/LogDelivery
```

4. Kemudian, buat file `logging.json` yang berisi konfigurasi logging Anda (berdasarkan salah satu dari tiga contoh berikut). Setelah Anda membuat file `logging.json`, Anda dapat menerapkan konfigurasi logging dengan menggunakan perintah `put-bucket-logging` berikut. Ganti *`example-s3-destination-bucket-logs`* dengan nama bucket tujuan (target) Anda.

```
aws s3api put-bucket-logging --bucket example-s3-destination-bucket-logs --bucket-
logging-status file://logging.json
```

> **Note**
>
> Alih-alih menggunakan perintah `put-bucket-logging` ini untuk menerapkan konfigurasi logging pada setiap bucket tujuan, Anda dapat menggunakan salah satu skrip bash yang disediakan di langkah berikutnya. Untuk menggunakan skrip tersebut, Anda harus membuat file `policy.json` dan `logging.json`, seperti yang dijelaskan dalam prosedur ini.

File `logging.json` ini adalah dokumen JSON di folder saat ini yang berisi konfigurasi pencatatan log. Jika bucket tujuan menggunakan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek, konfigurasi logging Anda tidak dapat berisi pemberian tujuan (target). Untuk informasi selengkapnya, lihat Izin untuk pengiriman log.

Example – **`logging.json`** tanpa pemberian tujuan (target)

File contoh `logging.json` berikut tidak berisi pemberian tujuan (target). Oleh karena itu, Anda dapat menerapkan konfigurasi ini ke bucket tujuan (target) yang menggunakan pengaturan yang diterapkan pemilik Bucket untuk Kepemilikan Objek.

```
{
```

```
    "LoggingEnabled": {
        "TargetBucket": "example-s3-destination-bucket-logs",
        "TargetPrefix": "example-s3-destination-bucket/"
     }
 }
```

Example – **`logging.json`** dengan pemberian tujuan (target)

File contoh `logging.json` berikut berisi pemberian tujuan (target).

Jika bucket tujuan menggunakan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek, Anda tidak dapat menyertakan pemberian tujuan (target) dalam konfigurasi PutBucketLogging Anda. Untuk informasi selengkapnya, lihat Izin untuk pengiriman log.

```
{
    "LoggingEnabled": {
        "TargetBucket": "example-s3-destination-bucket-logs",
        "TargetPrefix": "example-s3-destination-bucket/",
        "TargetGrants": [
             {
                "Grantee": {
                    "Type": "AmazonCustomerByEmail",
                    "EmailAddress": "user@example.com"
                 },
                "Permission": "FULL_CONTROL"
             }
         ]
     }
 }
```

Example - **`logging.json`** dengan format kunci objek log yang diatur ke waktu peristiwa S3

File `logging.json` berikut mengubah format kunci objek log ke waktu peristiwa S3. Untuk informasi selengkapnya tentang pengaturan format kunci objek log, lihat the section called “Bagaimana cara mengaktifkan log pengiriman?”

```
{
```

```
    "LoggingEnabled": {
        "TargetBucket": "example-s3-destination-bucket-logs",
        "TargetPrefix": "example-s3-destination-bucket/",
        "TargetObjectKeyFormat": {
            "PartitionedPrefix": {
                "PartitionDateSource": "EventTime"
            }
         }
    }
}
```

5. Gunakan salah satu skrip bash berikut untuk menambahkan pencatatan log akses untuk semua bucket dalam akun Anda. Ganti *example-s3-destination-bucket-logs* dengan nama bucket tujuan (target) Anda, dan ganti *us-west-2* dengan nama Wilayah tempat bucket Anda berada.

> **Note**
>
> Script ini hanya berfungsi jika semua bucket Anda berada di Wilayah yang sama. Jika Anda memiliki bucket di beberapa Wilayah, Anda harus menyesuaikan skrip-nya.

Example – Memberikan akses dengan kebijakan bucket dan menambahkan logging untuk bucket di akun Anda

```
loggingBucket='example-s3-destination-bucket-logs'
region='us-west-2'


# Create the logging bucket.
aws s3 mb s3://$loggingBucket --region $region

aws s3api put-bucket-policy --bucket $loggingBucket --policy file://policy.json

# List the buckets in this account.
buckets="$(aws s3 ls | awk '{print $3}')"

# Put a bucket logging configuration on each bucket.
for bucket in $buckets
    do
```

```
        # This if statement excludes the logging bucket.
        if [ "$bucket" != "$loggingBucket" ] ; then
            continue;
        fi
        printf '{
          "LoggingEnabled": {
            "TargetBucket": "%s",
            "TargetPrefix": "%s/"
        }
      }' "$loggingBucket" "$bucket"  > logging.json
      aws s3api put-bucket-logging --bucket $bucket --bucket-logging-status file://
logging.json
      echo "$bucket done"
  done

  rm logging.json

  echo "Complete"
```

Example – Memberikan akses dengan ACL bucket dan menambahkan logging untuk bucket di akun Anda

```
  loggingBucket='example-s3-destination-bucket-logs'
  region='us-west-2'


  # Create the logging bucket.
  aws s3 mb s3://$loggingBucket --region $region

  aws s3api put-bucket-acl --bucket $loggingBucket --grant-write URI=http://
acs.amazonaws.com/groups/s3/LogDelivery --grant-read-acp URI=http://
acs.amazonaws.com/groups/s3/LogDelivery

  # List the buckets in this account.
  buckets="$(aws s3 ls | awk '{print $3}')"

  # Put a bucket logging configuration on each bucket.
  for bucket in $buckets
      do
        # This if statement excludes the logging bucket.
        if [ "$bucket" != "$loggingBucket" ] ; then
            continue;
```

```
        fi
        printf '{
          "LoggingEnabled": {
            "TargetBucket": "%s",
            "TargetPrefix": "%s/"
        }
      }' "$loggingBucket" "$bucket"  > logging.json
      aws s3api put-bucket-logging --bucket $bucket --bucket-logging-status file://
logging.json
      echo "$bucket done"
  done

  rm logging.json

  echo "Complete"
```

Memverifikasi penyiapan log akses server Anda

Setelah Anda mengaktifkan pencatatan akses server, lakukan langkah berikut:

- Akses bucket tujuan dan verifikasi bahwa file log sedang dikirim. Setelah log akses disiapkan, mungkin diperlukan waktu lebih dari satu jam agar semua permintaan dicatat dan dikirim dengan benar. Anda juga dapat memverifikasi pengiriman log secara otomatis dengan menggunakan metrik permintaan Amazon S3 dan menyiapkan CloudWatch alarm Amazon untuk metrik ini. Untuk informasi selengkapnya, lihat Memantau metrik dengan Amazon CloudWatch.
- Verifikasi bahwa Anda dapat membuka dan membaca isi file log.

Untuk informasi pemecahan masalah pencatatan akses server, lihat Memecahkan masalah pencatatan akses server.

Server Amazon S3 mengakses format log

Pencatatan akses server menyediakan catatan terperinci untuk permintaan yang dilakukan ke bucket Amazon S3 Anda. Anda dapat menggunakan log akses server untuk tujuan berikut:

- Melakukan audit keamanan dan akses
- Belajar tentang basis pelanggan Anda
- Memahami tagihan Amazon S3 Anda

Bagian ini menjelaskan format dan detail lain tentang file log akses server Amazon S3.

Berkas log akses server terdiri atas urutan catatan log yang dibatasi di baris baru. Setiap catatan log mewakili satu permintaan dan terdiri dari bidang dengan ruang terbatas.

Berikut ini adalah contoh log yang terdiri dari lima catatan log.

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 DOC-EXAMPLE-BUCKET1 [06/Feb/2019:00:00:38 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be 3E57427F3EXAMPLE
 REST.GET.VERSIONING - "GET /DOC-EXAMPLE-BUCKET1?versioning HTTP/1.1" 200 - 113 - 7 -
 "-" "S3Console/0.4" - s9lzHYrFp76ZVxRcpX9+5cjAnEH2ROuNkd2BHfIa6UkFVdtjf5mKR3/eTPFvsiP/
XV/VLi31234= SigV4 ECDHE-RSA-AES128-GCM-SHA256 AuthHeader DOC-EXAMPLE-BUCKET1.s3.us-
west-1.amazonaws.com TLSV1.2 arn:aws:s3:us-west-1:123456789012:accesspoint/example-AP
 Yes
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 DOC-EXAMPLE-BUCKET1 [06/Feb/2019:00:00:38 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be 891CE47D2EXAMPLE
 REST.GET.LOGGING_STATUS - "GET /DOC-EXAMPLE-BUCKET1?logging HTTP/1.1" 200 -
 242 - 11 - "-" "S3Console/0.4" - 9vKBE6vMhrNiWHZmb2L0mXOcqPGzQOI5XLnCtZNPxev+Hf
+7tpT6sxDwDty4LHBUOZJG96N1234= SigV4 ECDHE-RSA-AES128-GCM-SHA256 AuthHeader DOC-
EXAMPLE-BUCKET1.s3.us-west-1.amazonaws.com TLSV1.2 - -
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 DOC-EXAMPLE-BUCKET1 [06/Feb/2019:00:00:38 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be A1206F460EXAMPLE
 REST.GET.BUCKETPOLICY - "GET /DOC-EXAMPLE-BUCKET1?policy HTTP/1.1" 404
 NoSuchBucketPolicy 297 - 38 - "-" "S3Console/0.4" - BNaBsXZQQDbssi6xMBdBU2sLt
+Yf5kZDmeBUP35sFoKa3sLLeMC78iwEIWxs99CRUrbS4n11234= SigV4 ECDHE-RSA-AES128-GCM-SHA256
 AuthHeader DOC-EXAMPLE-BUCKET1.s3.us-west-1.amazonaws.com TLSV1.2 - Yes
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 DOC-EXAMPLE-BUCKET1 [06/Feb/2019:00:01:00 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be 7B4A0FABBEXAMPLE
 REST.GET.VERSIONING - "GET /DOC-EXAMPLE-BUCKET1?versioning HTTP/1.1" 200 -
 113 - 33 - "-" "S3Console/0.4" - Ke1bUcazaN1jWuUlPJaxF64cQVpUEhoZKEG/hmy/gijN/
I1DeWqDfFvnpybfEseEME/u7ME1234= SigV4 ECDHE-RSA-AES128-GCM-SHA256 AuthHeader DOC-
EXAMPLE-BUCKET1.s3.us-west-1.amazonaws.com TLSV1.2 - -
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 DOC-EXAMPLE-BUCKET1 [06/Feb/2019:00:01:57 +0000] 192.0.2.3
 79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
 DD6CC733AEXAMPLE REST.PUT.OBJECT s3-dg.pdf "PUT /DOC-EXAMPLE-BUCKET1/
s3-dg.pdf HTTP/1.1" 200 - - 4406583 41754 28 "-" "S3Console/0.4" -
 10S62Zv81kBW7BB6SX4XJ48o6kpcl6LPwEoizZQQxJd5qDSCTLX0TgS37kYUBKQW3+bPdrg1234= SigV4
```

```
ECDHE-RSA-AES128-SHA AuthHeader DOC-EXAMPLE-BUCKET1.s3.us-west-1.amazonaws.com TLSV1.2
- Yes
```

> **Note**
>
> Bidang apa pun dapat diatur menjadi - untuk menunjukkan bahwa data tidak diketahui atau tidak tersedia, atau bahwa bidang tidak berlaku untuk permintaan ini.

Topik

- Bidang catatan log
- Pencatatan Tambahan untuk operasi penyalinan
- Informasi log akses kustom
- Pertimbangan pemrograman untuk format log akses server yang dapat diperluas

Bidang catatan log

Daftar berikut menjelaskan bidang catatan log.

Pemilik Bucket

ID pengguna resmi dari pemilik bucket sumber. ID pengguna kanonik adalah bentuk lain dari ID. Akun AWS Untuk informasi selengkapnya tentang ID pengguna kanonik, lihat Akun AWS pengidentifikasi di Referensi Umum AWS. Untuk informasi tentang cara menemukan ID pengguna kanonik untuk akun Anda, lihat Menemukan ID Pengguna kanonik Akun AWS Anda.

Entri contoh

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

Bucket

Nama bucket tempat permintaan diproses untuk dibandingkan. Jika sistem menerima permintaan yang salah bentuknya dan tidak dapat menentukan bucket, permintaan tidak akan muncul di log akses server apa pun.

Entri contoh

```
DOC-EXAMPLE-BUCKET1
```

Waktu

Waktu pada saat permintaan diterima; tanggal dan waktu ini berada dalam Waktu Universal Terkoordinasi (UTC). Formatnya, menggunakan terminologi `strftime()`, yaitu sebagai berikut: `[%d/%b/%Y:%H:%M:%S %z]`

Entri contoh

```
[06/Feb/2019:00:00:38 +0000]
```

IP Jarak Jauh

Alamat IP yang jelas dari pemohon. Proxy perantara dan firewall mungkin mengaburkan alamat IP aktual perangkat yang membuat permintaan.

Entri contoh

```
192.0.2.3
```

Peminta

ID pengguna kanonik pemohon, atau `-` untuk permintaan yang tidak terautentikasi. Jika pemohon adalah pengguna IAM, bidang ini mengembalikan nama pengguna IAM pemohon bersama dengan Pengguna root akun AWS yang dimiliki pengguna IAM. Pengidentifikasi ini sama dengan yang digunakan untuk tujuan kontrol akses.

Entri contoh

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

Jika pemohon menggunakan peran yang diasumsikan, bidang ini mengembalikan peran IAM yang diasumsikan.

Entri contoh

```
arn:aws:sts::123456789012:assumed-role/roleName/test-role
```

ID Permintaan

String yang dibuat oleh Amazon S3 untuk mengidentifikasi setiap permintaan secara unik.

Entri contoh

```
3E57427F33A59F07
```

Operasi

Operasi yang tercantum di sini dinyatakan sebagai `SOAP.operation`, `REST.HTTP_method.resource_type`, `WEBSITE.HTTP_method.resource_type`, atau `BATCH.DELETE.OBJECT`, atau `S3.action.resource_type` untuk Siklus hidup dan pencatatan.

Entri contoh

```
REST.PUT.OBJECT
```

Kunci

Kunci (nama objek) bagian dari permintaan.

Entri contoh

```
/photos/2019/08/puppy.jpg
```

Request-URI

Bagian `Request-URI` dari pesan permintaan HTTP.

Entri Contoh

```
"GET /DOC-EXAMPLE-BUCKET1/photos/2019/08/puppy.jpg?x-foo=bar HTTP/1.1"
```

Status HTTP

Kode status HTTP numerik dari respons.

Entri contoh

```
200
```

Kode Kesalahan

Amazon S3 Kode kesalahan, atau - jika tidak terjadi kesalahan.

Entri contoh

```
NoSuchBucket
```

Byte Dikirim

Jumlah byte respons yang dikirim, tidak termasuk overhead protokol HTTP, atau - jika nol.

Entri contoh

```
2662992
```

Ukuran Objek

Ukuran total objek yang dimaksud.

Entri contoh

```
3462992
```

Total Waktu

Jumlah milidetik permintaan dalam proses pengiriman dari perspektif server. Nilai ini diukur sejak permintaan Anda diterima hingga byte terakhir respons dikirimkan. Pengukuran yang dibuat dari perspektif klien mungkin lebih lama oleh sebab latensi jaringan.

Entri contoh

```
70
```

Waktu Perputaran

Jumlah milidetik yang dihabiskan Amazon S3 untuk memproses permintaan Anda. Nilai ini diukur sejak byte terakhir permintaan Anda diterima hingga byte pertama respons dikirim.

Entri contoh

```
10
```

Referer

Nilai dari header HTTP `Referer`, jika ada. Agen pengguna HTTP (misalnya, browser) biasanya mengatur header ini ke URL dari halaman penautan atau penyematan saat membuat permintaan.

Entri contoh

```
"http://www.example.com/webservices"
```

User-Agent

Nilai dari header HTTP `User-Agent`.

Entri contoh

```
"curl/7.15.1"
```

ID Versi

ID versi dalam permintaan, atau `-` jika operasi tidak mengambil `versionId` parameter.

Entri contoh

```
3HL4kqtJvjVBH40Nrjfkd
```

Id Host

ID permintaan `x-amz-id-2` yang diperluas atau Amazon S3.

Entri contoh

```
s9lzHYrFp76ZVxRcpX9+5cjAnEH2ROuNkd2BHfIa6UkFVdtjf5mKR3/eTPFvsiP/XV/VLi31234=
```

Versi Tanda Tangan

Versi tanda tangan, `SigV2` atau `SigV4`, yang digunakan untuk mengautentikasi permintaan atau `-` untuk permintaan tidak terautentikasi.

Entri contoh

```
SigV2
```

Rangkaian Penyandian

Cipher Lapisan Soket Aman (SSL) yang dinegosiasikan untuk permintaan HTTPS atau `-` untuk HTTP.

Entri contoh

```
ECDHE-RSA-AES128-GCM-SHA256
```

Jenis Autentikasi

Jenis autentikasi permintaan yang digunakan, `AuthHeader` untuk header autentikasi, `QueryString` untuk string kueri (URL yang ditandatangani sebelumnya) atau `-` untuk permintaan tidak terautentikasi.

Entri contoh

```
AuthHeader
```

Header Host

Titik akhir yang digunakan untuk terhubung ke Amazon S3.

Entri contoh

```
s3.us-west-2.amazonaws.com
```

Beberapa Wilayah sebelumnya mendukung titik akhir lama. Anda mungkin melihat titik akhir ini di log atau AWS CloudTrail log akses server Anda. Untuk informasi selengkapnya, lihat Titik akhir warisan. Untuk daftar lengkap titik akhir dan Wilayah Amazon S3, lihat Titik akhir dan kuota Amazon S3 di Referensi Umum Amazon Web Services.

Versi TLS

Versi Keamanan Lapisan Pengangkutan (TLS) yang dinegosiasikan oleh klien. Nilai tersebut adalah salah satu dari berikut ini:`TLSv1.1`, `TLSv1.2`, `TLSv1.3`; atau `-` jika TLS tidak digunakan.

Entri contoh

```
TLSv1.2
```

Titik Akses ARN

Amazon Resource Name (ARN) dari titik akses permintaan. Jika titik akses ARN mengalami kecacatan atau tidak digunakan, bidang akan berisi `-`. Untuk informasi lebih lanjut tentang titik akses, lihat Menggunakan titik akses. Untuk informasi selengkapnya tentang ARN, lihat Amazon Resource Name (ARN) di AWS Panduan Referensi.

Entri contoh

```
arn:aws:s3:us-east-1:123456789012:accesspoint/example-AP
```

aclRequired

String yang menunjukkan apakah permintaan memerlukan daftar kontrol akses (ACL) untuk otorisasi. Jika permintaan memerlukan ACL untuk otorisasi, stringnya adalah `Yes`. Jika tidak ada ACL yang diperlukan, stringnya adalah `-`. Untuk informasi selengkapnya tentang ACL, lihat Gambaran umum daftar kontrol akses (ACL). Untuk informasi selengkapnya tentang menggunakan bidang `aclRequired` untuk menonaktifkan ACL, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Entri contoh

```
Yes
```

Pencatatan Tambahan untuk operasi penyalinan

Sebuah operasi penyalinan melibatkan `GET` dan sebuah `PUT`. Karena alasan tersebut, kami mencatat dua catatan saat melakukan operasi penyalinan. Bagian sebelumnya menguraikan bidang yang terkait dengan bagian `PUT` dari operasi. Daftar berikut menjelaskan kolom dalam catatan yang berhubungan dengan bagian `GET` dari operasi penyalinan.

Pemilik Bucket

ID pengguna resmi dari bucket yang menyimpan objek yang disalin. ID pengguna kanonik adalah bentuk lain dari ID. Akun AWS Untuk informasi selengkapnya tentang ID pengguna kanonik, lihat

Akun AWS pengidentifikasi di Referensi Umum AWS. Untuk informasi tentang cara menemukan ID pengguna kanonik untuk akun Anda, lihat Menemukan ID Pengguna kanonik Akun AWS Anda.

Entri contoh

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

Bucket

Nama bucket yang menyimpan objek yang disalin.

Entri contoh

```
DOC-EXAMPLE-BUCKET1
```

Waktu

Waktu pada saat permintaan diterima; tanggal dan waktu ini berada dalam Waktu Universal Terkoordinasi (UTC). Formatnya, menggunakan terminologi `strftime()`, yaitu sebagai berikut: `[%d/%B/%Y:%H:%M:%S %z]`

Entri contoh

```
[06/Feb/2019:00:00:38 +0000]
```

IP Jarak Jauh

Alamat IP yang jelas dari pemohon. Proxy perantara dan firewall mungkin mengaburkan alamat IP aktual perangkat yang membuat permintaan.

Entri contoh

```
192.0.2.3
```

Peminta

ID pengguna kanonik pemohon, atau `-` untuk permintaan yang tidak terautentikasi. Jika pemohon adalah pengguna IAM, bidang ini akan mengembalikan nama pengguna IAM pemohon bersama dengan Pengguna root akun AWS yang dimiliki pengguna IAM. Pengidentifikasi ini sama dengan yang digunakan untuk tujuan kontrol akses.

Entri contoh

```
79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be
```

Jika pemohon menggunakan peran yang diasumsikan, bidang ini mengembalikan peran IAM yang diasumsikan.

Entri contoh

```
arn:aws:sts::123456789012:assumed-role/roleName/test-role
```

ID Permintaan

String yang dibuat oleh Amazon S3 untuk mengidentifikasi setiap permintaan secara unik.

Entri contoh

```
3E57427F33A59F07
```

Operasi

Operasi yang tercantum di sini dinyatakan sebagai `SOAP.operation`, `REST.HTTP_method.resource_type`, `WEBSITE.HTTP_method.resource_type`, atau `BATCH.DELETE.OBJECT`.

Entri contoh

```
REST.COPY.OBJECT_GET
```

Kunci

Kunci (nama objek) dari objek yang disalin, atau `-` jika operasi tidak mengambil parameter kunci.

Entri contoh

```
/photos/2019/08/puppy.jpg
```

Request-URI

Bagian `Request-URI` dari pesan permintaan HTTP.

Entri contoh

```
"GET /DOC-EXAMPLE-BUCKET1/photos/2019/08/puppy.jpg?x-foo=bar"
```

Status HTTP

Kode status HTTP numerik dari bagian `GET` dari operasi penyalinan.

Entri contoh

```
200
```

Kode Kesalahan

Amazon S3 Kode kesalahan, dari bagian `GET` dari operasi penyalinan atau `-` jika tidak terjadi kesalahan.

Entri contoh

```
NoSuchBucket
```

Byte Dikirim

Jumlah byte respons yang dikirim, tidak termasuk overhead protokol HTTP, atau `-` jika nol.

Entri contoh

```
2662992
```

Ukuran Objek

Ukuran total objek yang dimaksud.

Entri contoh

```
3462992
```

Total Waktu

Jumlah milidetik permintaan dalam proses pengiriman dari perspektif server. Nilai ini diukur sejak permintaan Anda diterima hingga byte terakhir respons dikirimkan. Pengukuran yang dibuat dari perspektif klien mungkin lebih lama oleh sebab latensi jaringan.

Entri contoh

```
70
```

Waktu Perputaran

Jumlah milidetik yang dihabiskan Amazon S3 untuk memproses permintaan Anda. Nilai ini diukur sejak byte terakhir permintaan Anda diterima hingga byte pertama respons dikirim.

Entri contoh

```
10
```

Referer

Nilai dari header HTTP `Referer`, jika ada. Agen pengguna HTTP (misalnya, browser) biasanya mengatur header ini ke URL dari halaman penautan atau penyematan saat membuat permintaan.

Entri contoh

```
"http://www.example.com/webservices"
```

User-Agent

Nilai dari header HTTP `User-Agent`.

Entri contoh

```
"curl/7.15.1"
```

ID Versi

ID versi objek yang disalin, atau `-` jika header `x-amz-copy-source` tidak menentukan parameter `versionId` sebagai bagian dari sumber salinan.

Entri Contoh

```
3HL4kqtJvjVBH40Nrjfkd
```

Id Host

ID permintaan `x-amz-id-2` yang diperluas atau Amazon S3.

Entri contoh

```
s9lzHYrFp76ZVxRcpX9+5cjAnEH2ROuNkd2BHfIa6UkFVdtjf5mKR3/eTPFvsiP/XV/VLi31234=
```

Versi Tanda Tangan

Versi tanda tangan, `SigV2` atau `SigV4`, yang digunakan untuk mengautentikasi permintaan atau `-` untuk permintaan tidak terautentikasi.

Entri contoh

```
SigV4
```

Rangkaian Penyandian

Cipher Lapisan Soket Aman (SSL) yang dinegosiasikan untuk permintaan HTTPS atau `-` untuk HTTP.

Entri contoh

```
ECDHE-RSA-AES128-GCM-SHA256
```

Jenis Autentikasi

Jenis autentikasi permintaan yang digunakan: `AuthHeader` untuk header autentikasi, `QueryString` untuk string kueri (URL yang ditandatangani sebelumnya), atau untuk permintaan tidak terautentikasi `-`.

Entri contoh

```
AuthHeader
```

Header Host

Titik akhir yang digunakan untuk terhubung ke Amazon S3.

Entri contoh

```
s3.us-west-2.amazonaws.com
```

Beberapa Wilayah sebelumnya mendukung titik akhir lama. Anda mungkin melihat titik akhir ini di log atau AWS CloudTrail log akses server Anda. Untuk informasi selengkapnya, lihat Titik akhir warisan. Untuk daftar lengkap titik akhir dan Wilayah Amazon S3, lihat Titik akhir dan kuota Amazon S3 di Referensi Umum Amazon Web Services.

Versi TLS

Versi Keamanan Lapisan Pengangkutan (TLS) yang dinegosiasikan oleh klien. Nilai tersebut adalah salah satu dari berikut ini:`TLSv1.1`, `TLSv1.2`, `TLSv1.3`; atau `-` jika TLS tidak digunakan.

Entri contoh

```
TLSv1.2
```

Titik Akses ARN

Amazon Resource Name (ARN) dari titik akses permintaan. Jika titik akses ARN mengalami kecacatan atau tidak digunakan, bidang akan berisi `-`. Untuk informasi lebih lanjut tentang titik akses, lihat Menggunakan titik akses. Untuk informasi selengkapnya tentang ARN, lihat Amazon Resource Name (ARN) di AWS Panduan Referensi.

Entri contoh

```
arn:aws:s3:us-east-1:123456789012:accesspoint/example-AP
```

aclRequired

String yang menunjukkan apakah permintaan memerlukan daftar kontrol akses (ACL) untuk otorisasi. Jika permintaan memerlukan ACL untuk otorisasi, stringnya adalah `Yes`. Jika tidak ada ACL yang diperlukan, stringnya adalah `-`. Untuk informasi selengkapnya tentang ACL, lihat Gambaran umum daftar kontrol akses (ACL). Untuk informasi selengkapnya tentang menggunakan bidang `aclRequired` untuk menonaktifkan ACL, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Entri contoh

```
Yes
```

Informasi log akses kustom

Anda dapat menyertakan informasi kustom untuk disimpan dalam catatan log akses untuk permintaan. Untuk melakukannya, menambahkan parameter string kueri kustom ke URL untuk permintaan tersebut. Amazon S3 mengabaikan parameter query-string yang dimulai dengan`x-`, tetapi mencakup parameter tersebut dalam catatan akses untuk permintaan, sebagai bagian dari `Request-URI` bidang catatan log.

Contohnya, permintaan `GET` untuk `"s3.amazonaws.com/DOC-EXAMPLE-BUCKET1/photos/2019/08/puppy.jpg?x-user=johndoe"` bekerja dengan cara yang sama seperti permintaan untuk `"s3.amazonaws.com/DOC-EXAMPLE-BUCKET1/photos/2019/08/puppy.jpg"`, kecuali bahwa string `"x-user=johndoe"` termasuk dalam bidang `Request-URI` untuk catatan log terkait. Fungsi ini tersedia di antarmuka REST saja.

Pertimbangan pemrograman untuk format log akses server yang dapat diperluas

Dari waktu ke waktu, kami dapat memperluas format catatan log akses dengan menambahkan bidang baru pada akhir setiap baris. Oleh karena itu, pastikan kode apa pun yang menguraikan pencatatan akses server dapat menangani bidang jejak yang mungkin tidak dipahaminya.

Menghapus file log Amazon S3

Bucket Amazon S3 dengan pencatatan akses server diaktifkan dapat mengakumulasi banyak objek log server dari waktu ke waktu. Aplikasi Anda mungkin memerlukan log akses ini untuk periode tertentu setelah pembuatan, dan setelah itu, Anda mungkin ingin menghapusnya. Anda dapat menggunakan konfigurasi siklus aktif Amazon S3 untuk menetapkan aturan sehingga Amazon S3 secara otomatis membuat antrean objek ini untuk dihapus di akhir masa aktifnya.

Anda dapat menentukan konfigurasi siklus aktif untuk bagian objek dalam bucket S3 Anda dengan menggunakan prefiks bersama. Jika Anda menentukan prefiks di konfigurasi pencatatan log akses server Anda, Anda dapat mengatur aturan konfigurasi siklus aktif untuk menghapus objek log yang memiliki prefiks tersebut.

Misalnya, anggaplah objek log Anda memiliki prefiks `logs/`. Anda dapat mengatur aturan konfigurasi siklus aktif untuk menghapus semua objek dalam bucket yang memiliki prefiks `logs/` setelah periode waktu tertentu.

Untuk informasi lebih lanjut tentang konfigurasi siklus aktif, lihat Mengelola siklus hidup penyimpanan Anda.

Untuk informasi selengkapnya tentang pencatatan akses server, lihat Pencatatan permintaan dengan pencatatan akses server.

Menggunakan log akses server Amazon S3 untuk mengidentifikasi permintaan

Anda dapat mengidentifikasi permintaan Amazon S3 dengan pencatatan akses server Amazon S3.

> **Note**
>
> - Untuk mengidentifikasi permintaan Amazon S3, sebaiknya gunakan peristiwa AWS CloudTrail data alih-alih log akses server Amazon S3. CloudTrail peristiwa data lebih mudah diatur dan berisi informasi lebih lanjut. Untuk informasi selengkapnya, lihat Mengidentifikasi permintaan Amazon S3 menggunakan CloudTrail.
> - Bergantung pada berapa banyak permintaan akses yang Anda dapatkan, menganalisis log Anda mungkin memerlukan lebih banyak sumber daya atau waktu daripada menggunakan peristiwa CloudTrail data.

Topik

- Melakukan kueri pencatatan akses untuk permintaan dengan menggunakan Amazon Athena
- Mengidentifikasi permintaan Signature Version 2 menggunakan pencatatan akses Amazon S3
- Mengidentifikasi permintaan akses objek dengan menggunakan pencatatan akses Amazon S3

Melakukan kueri pencatatan akses untuk permintaan dengan menggunakan Amazon Athena

Anda dapat mengidentifikasi permintaan Amazon S3 dengan log akses Amazon S3 menggunakan Amazon Athena.

Amazon S3 menyimpan log akses server sebagai objek dalam bucket S3. Sering kali lebih mudah menggunakan alat yang dapat menganalisis log di Amazon S3. Athena mendukung analisis S3 Object dan dapat digunakan untuk mencari log akses Amazon S3.

Example

Contoh berikut menunjukkan bagaimana Anda dapat melakukan kueri log akses server Amazon S3 di Amazon Athena. Ganti yang *user input placeholders* digunakan dalam contoh berikut dengan informasi Anda sendiri.

> **Note**
>
> Untuk menentukan lokasi Amazon S3 dalam kueri Athena, Anda harus memberikan URI S3 untuk bucket tempat log Anda dikirimkan. URI ini harus menyertakan nama bucket dan prefiks dalam format berikut: `s3://example-s3-bucket1-logs/prefix/`

1. Buka konsol Athena di https://console.aws.amazon.com/athena/.
2. Dalam Editor Kueri, jalankan perintah yang mirip dengan berikut ini. Ganti *s3_access_logs_db* dengan nama yang ingin Anda berikan ke basis data Anda.

```
CREATE DATABASE s3_access_logs_db
```

> **Note**
>
> Ini adalah praktik terbaik untuk membuat database yang Wilayah AWS sama dengan bucket S3 Anda.

3. Dalam Query Editor, jalankan perintah yang mirip dengan perintah berikut untuk membuat skema tabel di basis data yang Anda buat di langkah 2. Ganti *s3_access_logs_db.mybucket_logs* dengan nama yang ingin Anda berikan ke meja Anda. Nilai jenis data `STRING` dan `BIGINT` adalah properti log akses. Anda dapat mencari properti ini di Athena. Untuk `LOCATION`, masukkan bucket S3 dan jalur prefiks seperti yang disebutkan sebelumnya.

```
CREATE EXTERNAL TABLE `s3_access_logs_db.mybucket_logs`(
  `bucketowner` STRING,
  `bucket_name` STRING,
  `requestdatetime` STRING,
  `remoteip` STRING,
  `requester` STRING,
  `requestid` STRING,
  `operation` STRING,
  `key` STRING,
```

```
  `request_uri` STRING,
  `httpstatus` STRING,
  `errorcode` STRING,
  `bytessent` BIGINT,
  `objectsize` BIGINT,
  `totaltime` STRING,
  `turnaroundtime` STRING,
  `referrer` STRING,
  `useragent` STRING,
  `versionid` STRING,
  `hostid` STRING,
  `sigv` STRING,
  `ciphersuite` STRING,
  `authtype` STRING,
  `endpoint` STRING,
  `tlsversion` STRING,
  `accesspointarn` STRING,
  `aclrequired` STRING)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
  'input.regex'='([^ ]*) ([^ ]*) \\[(.*?)\\] ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)
 ([^ ]*) (\"[^\"]*\"|-) (-|[0-9]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)
 (\"[^\"]*\"|-) ([^ ]*)(?: ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)
 ([^ ]*))?.*$')
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  's3://DOC-EXAMPLE-BUCKET1-logs/prefix/'
```

4. Dalam panel navigasi, pada Basis Data, pilih basis data Anda.
5. Pada Tabel, pilih Tabel pratinjau di sebelah nama tabel Anda.

   Di panel Hasil, Anda akan melihat data dari log akses server, seperti `bucketowner`, `bucket`, `requestdatetime`, dan sebagainya. Ini berarti Anda berhasil membuat tabel Athena. Sekarang Anda dapat mengkueri log akses server Amazon S3.

Example — Menunjukkan siapa yang menghapus objek dan kapan (tanda waktu, alamat IP, dan pengguna IAM)

```
SELECT requestdatetime, remoteip, requester, key
FROM s3_access_logs_db.mybucket_logs
WHERE key = 'images/picture.jpg' AND operation like '%DELETE%';
```

Example — Menunjukkan semua operasi yang dilakukan oleh pengguna IAM

```
SELECT *
FROM s3_access_logs_db.mybucket_logs
WHERE requester='arn:aws:iam::123456789123:user/user_name';
```

Example — Menunjukkan semua operasi yang dilakukan pada objek dalam periode waktu tertentu

```
SELECT *
FROM s3_access_logs_db.mybucket_logs
WHERE Key='prefix/images/picture.jpg'
    AND parse_datetime(requestdatetime,'dd/MMM/yyyy:HH:mm:ss Z')
    BETWEEN parse_datetime('2017-02-18:07:00:00','yyyy-MM-dd:HH:mm:ss')
    AND parse_datetime('2017-02-18:08:00:00','yyyy-MM-dd:HH:mm:ss');
```

Example — Menunjukkan berapa banyak data yang dipindahkan ke alamat IP tertentu dalam periode waktu tertentu

```
SELECT coalesce(SUM(bytessent), 0) AS bytessenttotal
FROM s3_access_logs_db.mybucket_logs
WHERE remoteip='192.0.2.1'
AND parse_datetime(requestdatetime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2022-06-01','yyyy-MM-dd')
AND parse_datetime('2022-07-01','yyyy-MM-dd');
```

> **Note**
>
> Untuk mengurangi waktu dalam mempertahankan pencatatan, Anda dapat membuat konfigurasi Siklus Hidup S3 untuk bucket pencatatan akses server Anda. Buat aturan konfigurasi siklus aktif untuk menghapus file log secara berkala. Hal ini mengurangi jumlah data yang dianalisis oleh Athena untuk setiap kueri. Untuk informasi selengkapnya, lihat Menyetel konfigurasi siklus hidup pada bucket.

Mengidentifikasi permintaan Signature Version 2 menggunakan pencatatan akses Amazon S3

Dukungan Amazon S3 untuk Signature Version 2 akan dinonaktifkan (dihentikan). Setelah itu, Amazon S3 tidak akan menerima lagi permintaan yang menggunakan Signature Version 2, dan semua permintaan harus menggunakan penandatanganan Signature Version 4. Anda dapat mengidentifikasi permintaan akses Signature Version 2 menggunakan log akses Amazon S3.

> **Note**
>
> Untuk mengidentifikasi permintaan Signature Version 2, sebaiknya gunakan peristiwa AWS CloudTrail data, bukan log akses server Amazon S3. CloudTrail peristiwa data lebih mudah diatur dan berisi lebih banyak informasi daripada log akses server. Untuk informasi selengkapnya, lihat Mengidentifikasi permintaan Amazon S3 Signature Version 2 dengan menggunakan CloudTrail.

Example — Menunjukkan semua peminta yang mengirimkan lalu lintas Signature Version 2

```
SELECT requester, sigv, Count(sigv) as sigcount
FROM s3_access_logs_db.mybucket_logs
GROUP BY requester, sigv;
```

Mengidentifikasi permintaan akses objek dengan menggunakan pencatatan akses Amazon S3

Anda dapat menggunakan kueri di log akses server Amazon S3 untuk mengidentifikasi permintaan akses objek Amazon S3, untuk operasi seperti `GET`, `PUT`, dan `DELETE`, lalu menemukan informasi lebih lanjut tentang permintaan tersebut.

Contoh kueri Amazon Athena berikut menunjukkan cara mendapatkan semua permintaan objek `PUT` untuk Amazon S3 dari log akses server.

Example — Menunjukkan semua peminta yang mengirimkan permintaan objek **`PUT`** dalam periode tertentu

```
SELECT bucket_name, requester, remoteip, key, httpstatus, errorcode, requestdatetime
FROM s3_access_logs_db
WHERE operation='REST.PUT.OBJECT' AND
parse_datetime(requestdatetime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2019-07-01:00:42:42','yyyy-MM-dd:HH:mm:ss')
AND
parse_datetime('2019-07-02:00:42:42','yyyy-MM-dd:HH:mm:ss')
```

Contoh kueri Amazon Athena berikut menunjukkan cara mendapatkan semua permintaan objek `GET` untuk Amazon S3 dari log akses server.

Example — Menunjukkan semua peminta yang mengirimkan permintaan objek **`GET`** dalam periode tertentu

```
SELECT bucket_name, requester, remoteip, key, httpstatus, errorcode, requestdatetime
FROM s3_access_logs_db
WHERE operation='REST.GET.OBJECT' AND
parse_datetime(requestdatetime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2019-07-01:00:42:42','yyyy-MM-dd:HH:mm:ss')
AND
parse_datetime('2019-07-02:00:42:42','yyyy-MM-dd:HH:mm:ss')
```

Contoh kueri Amazon Athena berikut menunjukkan cara mendapatkan semua permintaan anonim ke bucket S3 Anda dari log akses server.

Example — Menunjukkan semua pemohon anonim yang membuat permintaan ke bucket selama periode tertentu

```
SELECT bucket_name, requester, remoteip, key, httpstatus, errorcode, requestdatetime
FROM s3_access_logs_db.mybucket_logs
WHERE requester IS NULL AND
parse_datetime(requestdatetime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2019-07-01:00:42:42','yyyy-MM-dd:HH:mm:ss')
AND
parse_datetime('2019-07-02:00:42:42','yyyy-MM-dd:HH:mm:ss')
```

Kueri Amazon Athena berikut menunjukkan cara mengidentifikasi semua permintaan ke bucket S3 Anda yang memerlukan daftar kontrol akses (ACL) untuk otorisasi. Anda dapat menggunakan informasi ini untuk memigrasikan izin ACL tersebut ke kebijakan bucket yang sesuai dan menonaktifkan ACL. Setelah membuat kebijakan bucket ini, Anda dapat menonaktifkan ACL untuk bucket ini. Untuk informasi selengkapnya tentang menonaktifkan ACL, lihat Prasyarat untuk menonaktifkan ACL.

Example — Identifikasi semua permintaan yang memerlukan ACL untuk otorisasi

```
SELECT bucket_name, requester, key, operation, aclrequired, requestdatetime
FROM s3_access_logs_db
WHERE aclrequired = 'Yes' AND
parse_datetime(requestdatetime,'dd/MMM/yyyy:HH:mm:ss Z')
BETWEEN parse_datetime('2022-05-10:00:00:00','yyyy-MM-dd:HH:mm:ss')
AND parse_datetime('2022-08-10:00:00:00','yyyy-MM-dd:HH:mm:ss')
```

Note

- Anda dapat mengubah rentang tanggal sesuai kebutuhan.
- Contoh kueri ini juga dapat berguna untuk pemantauan keamanan. Anda dapat meninjau hasil untuk panggilan `PutObject` atau `GetObject` dari alamat IP yang tidak terduga atau tidak sah atau pemohon, dan untuk mengidentifikasi permintaan anonim ke bucket Anda.
- Kueri ini hanya mengambil informasi dari waktu saat pencatatan log diaktifkan.
- Jika Anda menggunakan AWS CloudTrail log, lihatMengidentifikasi akses ke objek S3 dengan menggunakan CloudTrail.

Memantau metrik dengan Amazon CloudWatch

CloudWatch Metrik Amazon untuk Amazon S3 dapat membantu Anda memahami dan meningkatkan kinerja aplikasi yang menggunakan Amazon S3. Ada beberapa cara yang dapat Anda gunakan CloudWatch dengan Amazon S3.

Metrik penyimpanan harian untuk bucket

Pantau penyimpanan bucket menggunakan CloudWatch, yang mengumpulkan dan memproses data penyimpanan dari Amazon S3 menjadi metrik harian yang dapat dibaca. Metrik penyimpanan untuk Amazon S3 ini dilaporkan satu kali sehari dan diberikan kepada semua pelanggan tanpa biaya tambahan.

Metrik permintaan

Memantau permintaan Amazon S3 untuk mengidentifikasi dan menindaklanjuti masalah operasional dengan cepat. Metrik tersedia dengan interval 1 menit setelah beberapa latensi pemrosesan. CloudWatch Metrik ini ditagih dengan tarif yang sama dengan metrik CloudWatch khusus Amazon. Untuk informasi tentang CloudWatch harga, lihat CloudWatch harga Amazon. Untuk mempelajari cara memilih mendapatkan metrik ini, lihat CloudWatch konfigurasi metrik.

Saat diaktifkan, metrik permintaan dilaporkan untuk semua operasi objek. Secara default, metrik 1 menit ini tersedia di tingkat bucket Amazon S3. Anda juga dapat menentukan filter untuk metrik menggunakan prefiks bersama, tag objek, atau titik akses:

- Titik akses–Titik akses diberi nama titik akhir jaringan yang dilampirkan ke bucket dan menyederhanakan pengelolaan akses data pada skala untuk set data bersama di S3. Dengan filter titik akses, Anda dapat memperoleh wawasan tentang penggunaan titik akses Anda. Untuk informasi lebih lanjut tentang titik akses, lihat Pemantauan dan pencatatan titik akses.
- Prefiks–Meskipun model data Amazon S3 adalah struktur datar, Anda dapat menggunakan prefiks untuk menyimpulkan hierarki. Prefiks mirip dengan nama direktori yang mengaktifkan Anda untuk membuat grup objek yang sama bersama-sama dalam bucket. Konsol S3 mendukung prefiks dengan konsep folder. Jika Anda memfilter menurut prefiks, objek yang memiliki prefiks yang sama akan disertakan dalam konfigurasi metrik. Untuk informasi selengkapnya tentang prefiks, lihat Organisasi objek menggunakan prefiks.
- Tag–Tag adalah pasangan nama nilai-kunci yang dapat Anda tambahkan ke objek. Tag membantu Anda untuk menemukan dan menata objek dengan mudah. Anda juga dapat menggunakan tag sebagai filter untuk konfigurasi metrik sehingga hanya objek dengan tag

tersebut yang disertakan dalam konfigurasi metrik. Untuk informasi selengkapnya tentang tag objek, lihat Mengategorikan penyimpanan Anda menggunakan tag.

Untuk menyelaraskan metrik ini ke aplikasi bisnis, alur kerja, atau organisasi internal tertentu, Anda dapat memfilter pada prefiks bersama, tag objek, atau titik akses.

Metrik replikasi

Metrik replikasi–Memantau jumlah total operasi API S3 yang masih menunggu replikasi, ukuran total objek yang menunggu replikasi, waktu replikasi maksimal ke tujuan Wilayah AWS, dan jumlah total operasi yang gagal replikasi. Aturan replikasi yang memiliki Kontrol Waktu Replikasi S3 (S3 RTC) atau metrik replikasi S3 yang diaktifkan akan menerbitkan metrik replikasi.

Untuk informasi lebih lanjut, lihat Memantau progres dengan metrik replikasi dan Notifikasi Peristiwa S3 atau Memenuhi persyaratan kepatuhan menggunakan Kontrol Waktu Replikasi S3 (S3 RTC).

Metrik Lensa Penyimpanan Amazon S3

Anda dapat memublikasikan metrik penggunaan dan aktivitas Lensa Penyimpanan S3 CloudWatch ke Amazon untuk membuat tampilan terpadu kesehatan operasional Anda di dasbor. CloudWatch Metrik Lensa Penyimpanan S3 tersedia di namespace `AWS/S3/Storage-Lens`. Opsi CloudWatch penerbitan tersedia untuk dasbor Lensa Penyimpanan S3 yang ditingkatkan ke metrik dan rekomendasi lanjutan. Anda dapat mengaktifkan opsi CloudWatch penerbitan untuk konfigurasi dasbor baru atau yang sudah ada di S3 Storage Lens.

Untuk informasi selengkapnya, lihat Pantau metrik S3 Storage Lens di CloudWatch.

Semua CloudWatch statistik disimpan untuk jangka waktu 15 bulan sehingga Anda dapat mengakses informasi historis dan mendapatkan perspektif yang lebih baik tentang kinerja aplikasi atau layanan web Anda. Untuk informasi selengkapnya CloudWatch, lihat Apa itu Amazon CloudWatch? di Panduan CloudWatch Pengguna Amazon. Anda mungkin memerlukan beberapa konfigurasi tambahan untuk CloudWatch alarm Anda, tergantung pada kasus penggunaan Anda. Misalnya, Anda dapat menggunakan ekspresi matematika metrik untuk membuat alarm. Untuk informasi selengkapnya, lihat Menggunakan CloudWatch metrik, Menggunakan matematika metrik, Menggunakan CloudWatch alarm Amazon, dan Membuat CloudWatch alarm berdasarkan ekspresi matematika metrik di CloudWatch Panduan Pengguna Amazon.

Pengiriman CloudWatch metrik upaya terbaik

CloudWatch metrik disampaikan atas dasar upaya terbaik. Sebagian besar permintaan untuk objek Amazon S3 yang memiliki metrik permintaan menghasilkan titik data yang dikirim. CloudWatch

Kelengkapan dan ketepatan waktu metrik tidak dijamin. Titik data untuk permintaan tertentu mungkin dikembalikan dengan stempel waktu yang lebih lambat daripada ketika permintaan tersebut sebenarnya diproses. Titik data selama satu menit mungkin tertunda sebelum tersedia CloudWatch, atau mungkin tidak dikirimkan sama sekali. CloudWatch metrik permintaan memberi Anda gambaran tentang sifat lalu lintas terhadap bucket Anda dalam waktu hampir nyata. Ini tidak dimaksudkan untuk menjadi pencatatan akuntasi yang lengkap atas semua permintaan.

Berdasarkan sifat upaya terbaik dari fitur ini, laporan yang tersedia di Dasbor Manajemen Penagihan & Biaya mungkin menyertakan satu atau lebih permintaan akses yang tidak muncul dalam metrik bucket.

Untuk informasi selengkapnya, lihat topik berikut.

Topik

- Metrik dan dimensi
- Mengakses metrik CloudWatch
- CloudWatch konfigurasi metrik

Metrik dan dimensi

Metrik dan dimensi penyimpanan yang dikirimkan Amazon S3 ke CloudWatch Amazon tercantum dalam tabel berikut.

Pengiriman CloudWatch metrik upaya terbaik

CloudWatch metrik disampaikan atas dasar upaya terbaik. Sebagian besar permintaan untuk objek Amazon S3 yang memiliki metrik permintaan menghasilkan titik data yang dikirim. CloudWatch

Kelengkapan dan ketepatan waktu metrik tidak dijamin. Titik data untuk permintaan tertentu mungkin dikembalikan dengan stempel waktu yang lebih lambat daripada ketika permintaan tersebut sebenarnya diproses. Titik data selama satu menit mungkin tertunda sebelum tersedia CloudWatch, atau mungkin tidak dikirimkan sama sekali. CloudWatch metrik permintaan memberi Anda gambaran tentang sifat lalu lintas terhadap bucket Anda dalam waktu hampir nyata. Ini tidak dimaksudkan untuk menjadi pencatatan akuntasi yang lengkap atas semua permintaan.

Berdasarkan sifat upaya terbaik dari fitur ini, laporan yang tersedia di Dasbor Manajemen Penagihan & Biaya mungkin menyertakan satu atau lebih permintaan akses yang tidak muncul dalam metrik bucket.

Topik

- Metrik penyimpanan harian Amazon S3 untuk bucket di CloudWatch
- Metrik permintaan Amazon S3 di CloudWatch
- Metrik replikasi S3 di CloudWatch
- Metrik Lensa Penyimpanan S3 di CloudWatch
- Metrik permintaan Lambda Objek S3 di CloudWatch
- Amazon S3 pada metrik Outposts di CloudWatch
- Dimensi Amazon S3 di CloudWatch
- Dimensi replikasi S3 di CloudWatch
- Dimensi Lensa Penyimpanan S3 di CloudWatch
- Dimensi permintaan Lambda Objek S3 di CloudWatch

Metrik penyimpanan harian Amazon S3 untuk bucket di CloudWatch

Namespace `AWS/S3` mencakup metrik penyimpanan harian berikut untuk bucket.

Metrik	Deskripsi
`BucketSizeBytes`	Jumlah data dalam byte yang disimpan dalam bucket di kelas penyimpanan berikut: • S3 Standard (`STANDARD`) • S3 Intelligent-Tiering (`INTELLIGENT_TIERING`) • S3 Standard-Infrequent Access (`STANDARD_IA`) • Akses S3 Satu Zona Jarang () `ONEZONE_IA` • Reduced Redundancy Storage (RRS) (`REDUCED_REDUNDANCY`) • S3 Glacier Instant Retrieval (`GLACIER_IR`) • S3 Glacier Deep Archive (`DEEP_ARCHIVE`) • S3 Glacier Flexible Retrieval (`GLACIER`)

Metrik	Deskripsi
	• S3 Express One Zone (`EXPRESS_ONEZONE`) Nilai ini dihitung dengan menjumlahkan ukuran semua objek dan metadata (seperti nama bucket) dalam bucket (baik objek saat ini maupun noncurrent), termasuk ukuran semua bagian untuk semua unggahan multipart yang tidak lengkap ke bucket. Filter jenis penyimpanan yang valid: `StandardStorage`, `IntelligentTieringFAStorage`, `IntelligentTieringIAStorage`, `IntelligentTieringAAStorage`, `IntelligentTieringAIAStorage`, `IntelligentTieringDAAStorage`, `StandardIAStorage`, `StandardIASizeOverhead`, `StandardIAObjectOverhead`, `OneZoneIAStorage`, `OneZoneIASizeOverhead`, `ReducedRedundancyStorage`, `GlacierInstantRetrievalSizeOverhead` `GlacierInstantRetrievalStorage`, `GlacierStorage`, `GlacierStagingStorage`, `GlacierObjectOverhead`, `GlacierS3ObjectOverhead`, `DeepArchiveStorage`, `DeepArchiveObjectOverhead`, `DeepArchiveS3ObjectOverhead`, `DeepArchiveStagingStorage`, dan `ExpressOneZone` (lihat dimensi `StorageType`) Unit: Bita Statistik yang valid: Rata-rata

Metrik	Deskripsi
`NumberOfObjects`	Total jumlah penyimpanan objek dalam bucket tujuan umum untuk semua kelas penyimpanan. Nilai ini dihitung dengan menghitung semua objek dalam bucket, yang mencakup objek saat ini dan yang tidak berjalan, penanda hapus, dan jumlah total bagian untuk semua unggahan multibagian yang tidak lengkap ke bucket. Untuk bucket direktori dengan objek di kelas penyimpanan S3 Express One Zone, nilai ini dihitung dengan menghitung semua objek dalam bucket, tetapi tidak termasuk beberapa unggahan yang tidak lengkap ke bucket. Filter jenis penyimpanan yang valid: `AllStorageTypes` (lihat dimensi `StorageType`) Unit: Hitungan Statistik yang valid: Rata-rata

Metrik permintaan Amazon S3 di CloudWatch

Namespace `AWS/S3` menyertakan metrik permintaan berikut. Metrik ini mencakup permintaan yang tidak dapat ditagih (dalam kasus `GET` permintaan dari `CopyObject` dan Replikasi).

> **Note**
>
> Metrik permintaan Amazon S3 CloudWatch tidak didukung untuk bucket direktori.

Metrik	Deskripsi
`AllRequests`	Total jumlah permintaan HTTP yang dilakukan ke bucket Amazon S3, apa pun jenisnya. Jika Anda menggunakan konfigurasi metrik dengan filter, maka metrik ini hanya mengembalikan permintaan HTTP yang memenuhi persyaratan filter. Unit: Hitungan Statistik yang valid: Jumlah

Metrik	Deskripsi
`GetRequests`	Jumlah `GET` permintaan HTTP yang dibuat untuk objek dalam bucket Amazon S3. Ini tidak termasuk operasi daftar. Metrik ini ditambahkan untuk sumber setiap `CopyObject` permintaan. Unit: Hitungan Statistik yang valid: Jumlah **Note** Permintaan berorientasi daftar paginasi, seperti ListMultipartUploads,, ListParts, dan lainnya ListObjectVersions, tidak termasuk dalam metrik ini.
`PutRequests`	Jumlah `PUT` permintaan HTTP yang dibuat untuk objek dalam bucket Amazon S3. Metrik ini ditambahkan untuk tujuan setiap `CopyObject` permintaan. Unit: Hitungan Statistik yang valid: Jumlah
`DeleteRequests`	Jumlah `DELETE` permintaan HTTP yang dibuat untuk objek dalam bucket Amazon S3. Metrik ini juga mencakup DeleteObjectspermintaan. Metrik ini menunjukkan jumlah permintaan yang dibuat, bukan jumlah objek yang dihapus. Unit: Hitungan Statistik yang valid: Jumlah
`HeadRequests`	Jumlah permintaan HTTP `HEAD` yang dibuat ke bucket Amazon S3. Unit: Hitungan Statistik yang valid: Jumlah

Metrik	Deskripsi
`PostRequests`	Jumlah permintaan HTTP `POST` yang dibuat ke bucket Amazon S3. Unit: Hitungan Statistik yang valid: Jumlah **Note** DeleteObjectsdan SelectObjectContentpermintaan tidak termasuk dalam metrik ini.
`SelectRequests`	Jumlah SelectObjectContentpermintaan Amazon S3 yang dibuat untuk objek di bucket Amazon S3. Unit: Hitungan Statistik yang valid: Jumlah
`SelectBytesScanned`	Jumlah byte data yang dipindai dengan permintaan Amazon S3 di SelectObjectContentbucket Amazon S3. Unit: Bita Statistik yang valid: Rata-rata (byte per permintaan), Jumlah (byte per periode), Jumlah Sampel, Min, Maks (sama dengan p100), persentil apa pun antara p0,0 dan p99,9
`SelectBytesReturned`	Jumlah byte data yang dikembalikan dengan permintaan Amazon SelectObjectContentS3 di bucket Amazon S3. Unit: Bita Statistik yang valid: Rata-rata (byte per permintaan), Jumlah (byte per periode), Jumlah Sampel, Min, Maks (sama dengan p100), persentil apa pun antara p0,0 dan p99,9

Metrik	Deskripsi
`ListRequests`	Jumlah permintaan HTTP yang mencantumkan daftar isi bucket. Unit: Hitungan Statistik yang valid: Jumlah
`BytesDownloaded`	Jumlah byte yang diunduh untuk permintaan yang dibuat ke bucket Amazon S3, yang tanggapannya mencakup satu isi. Unit: Byte Statistik yang valid: Rata-rata (byte per permintaan), Jumlah (byte per periode), Jumlah Sampel, Min, Maks (sama dengan p100), persentil apa pun antara p0,0 dan p99,9
`BytesUploaded`	Jumlah byte yang diunggah untuk permintaan yang dibuat ke bucket Amazon S3, yang permintaannya mencakup satu isi. Unit: Bita Statistik yang valid: Rata-rata (byte per permintaan), Jumlah (byte per periode), Jumlah Sampel, Min, Maks (sama dengan p100), persentil apa pun antara p0,0 dan p99,9
`4xxErrors`	Jumlah permintaan kode status kesalahan klien HTTP 4 xx yang dibuat ke bucket Amazon S3 dengan nilai 0 atau 1. Statistik rata-rata menunjukkan tingkat kesalahan, dan statistik Jumlah menunjukkan jumlah jenis kesalahan tersebut, selama setiap periode. Unit: Hitungan Statistik yang valid: Rata-rata (laporan per permintaan), Jumlah (laporan per periode), Min, Maks, Jumlah Sampel

Metrik	Deskripsi
`5xxErrors`	Jumlah permintaan kode status kesalahan server HTTP 5 xx yang dibuat ke bucket Amazon S3 dengan nilai 0 atau 1. Statistik rata-rata menunjukkan tingkat kesalahan, dan statistik Jumlah menunjukkan jumlah jenis kesalahan tersebut, selama setiap periode. Unit: Hitungan Statistik yang valid: Rata-rata (laporan per permintaan), Jumlah (laporan per periode), Min, Maks, Jumlah Sampel
`FirstByteLatency`	Waktu per permintaan dari permintaan lengkap yang diterima oleh bucket Amazon S3 saat respons mulai dikembalikan. Unit: Milidetik Statistik yang valid: Rata-rata, Jumlah, Min, Maks (sama seperti p100), Jumlah Sampel, persentil apa pun antara p0,0 dan p100
`TotalRequestLatency`	Waktu per permintaan yang berlalu dari byte pertama yang diterima hingga byte terakhir yang dikirim ke bucket Amazon S3. Metrik ini mencakup waktu yang dibutuhkan untuk menerima isi permintaan dan mengirimkan isi respons, yang tidak termasuk dalam `FirstByteLatency` . Unit: Milidetik Statistik yang valid: Rata-rata, Jumlah, Min, Maks (sama seperti p100), Jumlah Sampel, persentil apa pun antara p0,0 dan p100

Metrik replikasi S3 di CloudWatch

Anda dapat memantau kemajuan replikasi dengan metrik replikasi S3 dengan melacak byte tertunda, operasi tertunda, dan latensi replikasi. Untuk informasi selengkapnya, lihat Memantau progres dengan metrik replikasi.

> **Note**
>
> Anda dapat mengaktifkan alarm untuk metrik replikasi Anda di Amazon. CloudWatch Saat Anda menyiapkan alarm untuk metrik replikasi Anda, tetapkan bidang Perawatan data yang hilang untuk Tangani data yang hilang sebagai diabaikan (jaga status alarm).

Metrik	Deskripsi
`ReplicationLatency`	Jumlah maksimum detik dimana tujuan replikasi Wilayah AWS berada di belakang sumber Wilayah AWS untuk aturan replikasi yang diberikan. Unit: detik Statistik yang valid: Maks
`BytesPendingReplication`	Jumlah total byte objek yang menunggu replikasi untuk aturan replikasi yang diberikan. Unit: Byte Statistik yang valid: Maks
`OperationsPendingReplication`	Jumlah operasi yang menunggu replikasi untuk aturan replikasi yang diberikan. Unit: Hitungan Statistik yang valid: Maks
`OperationsFailedReplication`	Jumlah operasi yang gagal untuk mereplikasi untuk aturan replikasi yang diberikan. Unit: Hitungan Statistik yang valid: Jumlah (jumlah total operasi yang gagal), Rata-rata (tingkat kegagalan), Jumlah Sampel (jumlah total operasi replikasi)

Metrik Lensa Penyimpanan S3 di CloudWatch

Anda dapat memublikasikan metrik penggunaan dan aktivitas Lensa Penyimpanan S3 CloudWatch ke Amazon untuk membuat tampilan terpadu kesehatan operasional Anda di dasbor. CloudWatch Metrik Lensa Penyimpanan S3 dipublikasikan ke `AWS/S3/Storage-Lens` namespace di. CloudWatch Opsi CloudWatch penerbitan tersedia untuk dasbor Lensa Penyimpanan S3 yang telah ditingkatkan ke metrik dan rekomendasi lanjutan.

Untuk daftar metrik Lensa Penyimpanan S3 yang dipublikasikan CloudWatch, lihat. Glosarium metrik Amazon S3 Storage Lens Untuk daftar lengkap dimensi, lihat Dimensi.

Metrik permintaan Lambda Objek S3 di CloudWatch

S3 Lambda Objek menyertakan metrik permintaan berikut.

Metrik	Deskripsi
`AllRequests`	Total jumlah permintaan HTTP yang dibuat ke bucket Amazon S3 dengan menggunakan Titik Akses Lambda Object. Unit: Hitungan Statistik yang valid: Jumlah
`GetRequests`	Jumlah permintaan HTTP `GET` yang dibuat untuk objek dengan menggunakan Titik Akses Lambda Object. Metrik ini tidak mencakup operasi daftar. Unit: Hitungan Statistik yang valid: Jumlah
`BytesUploaded`	Jumlah byte yang diunggah ke bucket Amazon S3 menggunakan Titik Akses Lambda Object, yang permintaannya menyertakan isi. Unit: Bita Statistik yang valid: Rata-rata (byte per permintaan), Jumlah (byte per periode), Jumlah Sampel, Min, Maks (sama dengan p100), persentil apa pun antara p0,0 dan p99,9

Metrik	Deskripsi
`PostRequests`	Jumlah permintaan HTTP `POST` yang dibuat ke bucket Amazon S3 dengan menggunakan Titik Akses Lambda Object. Unit: Hitungan Statistik yang valid: Jumlah
`PutRequests`	Jumlah permintaan HTTP `PUT` yang dibuat untuk objek dalam bucket Amazon S3 menggunakan Titik Akses Lambda Object. Unit: Hitungan Statistik yang valid: Jumlah
`DeleteRequests`	Jumlah permintaan HTTP `DELETE` yang dibuat untuk objek dalam bucket Amazon S3 menggunakan Titik Akses Lambda Objek. Metrik ini mencakup DeleteObjectspermintaan. Metrik ini menunjukkan jumlah permintaan yang dibuat, bukan jumlah objek yang dihapus. Unit: Hitungan Statistik yang valid: Jumlah
`BytesDownloaded`	Jumlah byte yang diunduh untuk permintaan yang dibuat ke bucket Amazon S3 menggunakan Titik Akses Lambda Object, yang tanggapan nya mencakup satu isi. Unit: Bita Statistik yang valid: Rata-rata (byte per permintaan), Jumlah (byte per periode), Jumlah Sampel, Min, Maks (sama dengan p100), persentil apa pun antara p0,0 dan p99,9

Metrik	Deskripsi
`FirstByteLatency`	Waktu per permintaan dari permintaan lengkap yang diterima oleh bucket Amazon S3 melalui Titik Akses Lambda Objek hingga saat respons mulai dikembalikan. Metrik ini tergantung pada waktu berjalan pada fungsi AWS Lambda untuk mengubah objek sebelum fungsi mengembalikan byte ke Titik Akses Lambda Objek. Unit: Milidetik Statistik yang valid: Rata-rata, Jumlah, Min, Maks (sama seperti p100), Jumlah Sampel, persentil apa pun antara p0,0 dan p100
`TotalRequestLatency`	Waktu per permintaan yang berlalu dari byte pertama yang diterima hingga byte terakhir yang dikirim ke Titik Akses Lambda Objek. Metrik ini mencakup waktu yang dibutuhkan untuk menerima isi permintaan dan mengirimkan isi respons, yang tidak termasuk dalam `FirstByteLatency` . Unit: Milidetik Statistik yang valid: Rata-rata, Jumlah, Min, Maks (sama seperti p100), Jumlah Sampel, persentil apa pun antara p0,0 dan p100
`HeadRequests`	Jumlah permintaan HTTP `HEAD` yang dibuat ke bucket Amazon S3 dengan menggunakan Titik Akses Lambda Objek. Unit: Hitungan Statistik yang valid: Jumlah
`ListRequests`	Jumlah permintaan HTTP `GET` yang mencantumkan daftar isi bucket Amazon S3. Metrik ini mencakup kedua operasi `ListObjects` dan `ListObjectsV2` . Unit: Hitungan Statistik yang valid: Jumlah

Metrik	Deskripsi
`4xxErrors`	Jumlah permintaan kode status kesalahan klien HTTP 4 xx yang dibuat ke bucket Amazon S3 dengan menggunakan Object Lambda Access Point dengan nilai 0 atau 1. Statistik rata-rata menunjukkan tingkat kesalahan, dan statistik Jumlah menunjukkan jumlah jenis kesalahan tersebut, selama setiap periode. Unit: Hitungan Statistik yang valid: Rata-rata (laporan per permintaan), Jumlah (laporan per periode), Min, Maks, Jumlah Sampel
`5xxErrors`	Jumlah permintaan kode status kesalahan server HTTP 5 xx yang dibuat ke bucket Amazon S3 dengan menggunakan Object Lambda Access Point dengan nilai 0 atau 1. Statistik rata-rata menunjukkan tingkat kesalahan, dan statistik Jumlah menunjukkan jumlah jenis kesalahan tersebut, selama setiap periode. Unit: Hitungan Statistik yang valid: Rata-rata (laporan per permintaan), Jumlah (laporan per periode), Min, Maks, Jumlah Sampel
`ProxiedRequests`	Jumlah permintaan HTTP ke Titik Akses Lambda Objek yang mengembalikan respons API Amazon S3 standar. (Permintaan tersebut tidak memiliki fungsi Lambda yang dikonfigurasi.) Unit: Hitungan Statistik yang valid: Jumlah
`InvokedLambda`	Jumlah permintaan HTTP ke S3 Object di mana fungsi Lambda dipanggil . Unit: Hitungan Statistik yang valid: Jumlah

Metrik	Deskripsi
`LambdaResponseRequests`	Jumlah permintaan `WriteGetObjectResponse` yang dibuat oleh fungsi Lambda. Metrik ini hanya berlaku untuk permintaan `GetObject`.
`LambdaResponse4xx`	Jumlah kesalahan klien HTTP 4 xx yang terjadi saat memanggil `WriteGetObjectResponse` dari fungsi Lambda. Metrik ini memberikan informasi yang sama seperti `4xxErrors`, tetapi hanya untuk panggilan `WriteGetObjectResponse`.
`LambdaResponse5xx`	Jumlah kesalahan server HTTP 5 xx yang terjadi saat memanggil `WriteGetObjectResponse` dari fungsi Lambda. Metrik ini memberikan informasi yang sama seperti `5xxErrors`, tetapi hanya untuk panggilan `WriteGetObjectResponse`.

Amazon S3 pada metrik Outposts di CloudWatch

Untuk daftar metrik CloudWatch yang digunakan untuk S3 pada bucket Outposts, lihat. CloudWatch metrik

Dimensi Amazon S3 di CloudWatch

Dimensi berikut digunakan untuk memfilter metrik Amazon S3.

Dimensi	Deskripsi
`BucketName`	Dimensi ini memfilter data yang Anda minta untuk bucket yang diidentifikasi saja.
`StorageType`	Dimensi ini memfilter data yang telah Anda simpan dalam bucket dengan jenis penyimpanan berikut: • `StandardStorage` –Jumlah byte yang digunakan untuk objek dalam kelas penyimpanan `STANDARD`. • `IntelligentTieringAAStorage` –Jumlah byte yang digunakan untuk objek di tingkat Archive Access dari kelas penyimpanan `INTELLIGENT_TIERING`.

Dimensi	Deskripsi
	• `IntelligentTieringAIAStorage` — Jumlah byte yang digunakan untuk objek di tingkat Archive Instant Access kelas penyimpanan `INTELLIGENT_TIERING` . • `IntelligentTieringDAAStorage` –Jumlah byte yang digunakan untuk objek di tingkat Deep Archive Access dari kelas penyimpanan `INTELLIGENT_TIERING` . • `IntelligentTieringFAStorage` –Jumlah byte yang digunakan untuk objek di tingkat Frequent Access dari kelas penyimpanan `INTELLIGENT_TIERING` . • `IntelligentTieringIAStorage` –Jumlah byte yang digunakan untuk objek di tingkat Infrequent Access dari kelas penyimpanan `INTELLIGENT_TIERING` . • `StandardIAStorage` — Jumlah byte yang digunakan untuk objek di kelas penyimpanan S3 Standard-Infrequent Access (). `STANDARD_IA` • `StandardIASizeOverhead` –Jumlah byte yang digunakan untuk objek berukuran lebih kecil dari 128 KB dalam kelas penyimpanan `STANDARD_IA` . • `IntAAObjectOverhead` –Untuk setiap objek dalam kelas penyimpanan `INTELLIGENT_TIERING` di tingkat Archive Access, S3 Glacier menambahkan 32 KB penyimpanan untuk indeks dan metadata terkait. Data ekstra ini diperluka n untuk mengidentifikasi dan memulihkan objek Anda. Anda dikenakan tarif S3 Glacier Flexible Retrieval untuk penyimpan an tambahan ini. • `IntAAS3ObjectOverhead` –Untuk setiap objek dalam kelas penyimpanan `INTELLIGENT_TIERING` di tingkat Archive Access, Amazon S3 menggunakan penyimpanan sebesar 8 KB untuk nama objek dan metadata lainnya. Anda dikenakan tarif Standar S3 untuk penyimpanan tambahan ini. • `IntDAAObjectOverhead` –Untuk setiap objek dalam kelas penyimpanan `INTELLIGENT_TIERING` di tingkat Deep Archive Access, S3 Glacier menambahkan penyimpanan

Dimensi	Deskripsi
	sebesar 32 KB untuk indeks dan metadata terkait. Data ekstra ini diperlukan untuk mengidentifikasi dan memulihkan objek Anda. Anda dikenakan tarif biaya S3 Glacier Deep Archive untuk penyimpanan tambahan ini. • `IntDAAS3ObjectOverhead` –Untuk setiap objek dalam kelas penyimpanan `INTELLIGENT_TIERING` di tingkat Deep Archive Access, Amazon S3 menambahkan penyimpan an sebesar 8 KB untuk indeks dan metadata terkait. Data ekstra ini diperlukan untuk mengidentifikasi dan memulihka n objek Anda. Anda dikenakan tarif Standar S3 untuk penyimpanan tambahan ini. • `OneZoneIAStorage` –Jumlah byte yang digunakan untuk objek di kelas penyimpanan S3 One Zone-Infrequent Access (`ONEZONE_IA`). • `OneZoneIASizeOverhead` –Jumlah byte yang digunakan untuk objek berukuran lebih kecil dari 128 KB dalam kelas penyimpanan `ONEZONE_IA` . • `ReducedRedundancyStorage` –Jumlah byte yang digunakan untuk objek dalam kelas Reduced Redundancy Storage (RRS). • `GlacierInstantRetrievalSizeOverhead` –Jumlah byte yang digunakan untuk objek berukuran lebih kecil dari 128 KB di kelas penyimpanan Instant Retrieval S3 Glacier Instant Retrieval. • `GlacierInstantRetrievalStorage` –Jumlah byte yang digunakan untuk objek dalam kelas penyimpanan S3 Glacier Instant Retrieval. • `GlacierStorage` –Jumlah byte yang digunakan untuk objek dalam kelas penyimpanan S3 Glacier Flexible Retrieval. • `GlacierStagingStorage` — Jumlah byte yang digunakan untuk bagian objek multibagian sebelum permintaa n `CompleteMultipartUpload` diselesaikan pada objek dalam kelas penyimpanan S3 Glacier Flexible Retrieval.

Dimensi	Deskripsi
	• `GlacierObjectOverhead` –Untuk setiap objek yang diarsipkan, S3 Glacier menambahkan penyimpanan 32 KB untuk indeks dan metadata terkait. Data ekstra ini diperluka n untuk mengidentifikasi dan memulihkan objek Anda. Anda dikenakan tarif S3 Glacier Flexible Retrieval untuk penyimpan an tambahan ini. • `GlacierS3ObjectOverhead` –Untuk setiap objek yang diarsipkan ke S3 Glacier Flexible Retrieval, Amazon S3 menggunakan penyimpanan 8 KB untuk nama objek dan metadata lainnya. Anda dikenakan tarif Standar S3 untuk penyimpanan tambahan ini. • `DeepArchiveStorage` –Jumlah byte yang digunakan untuk objek dalam kelas penyimpanan S3 Glacier Deep Archive. • `DeepArchiveObjectOverhead` –Untuk setiap objek yang diarsipkan, S3 Glacier menambahkan penyimpanan 32 KB untuk indeks dan metadata terkait. Data ekstra ini diperlukan untuk mengidentifikasi dan memulihkan objek Anda. Anda dikenakan tarif biaya S3 Glacier Deep Archive untuk penyimpanan tambahan ini. • `DeepArchiveS3ObjectOverhead` –Untuk setiap objek yang diarsipkan ke S3 Glacier Deep Archive, Amazon S3 menggunakan penyimpanan 8 KB untuk nama objek dan metadata lainnya. Anda dikenakan tarif Standar S3 untuk penyimpanan tambahan ini. • `DeepArchiveStagingStorage` –Jumlah byte yang digunakan untuk bagian objek Multibagian sebelum permintaa n `CompleteMultipartUpload` diselesaikan pada objek dalam kelas penyimpanan S3 Glacier Deep Archive. • `ExpressOneZone` –Jumlah byte yang digunakan untuk objek dalam kelas penyimpanan S3 Express One Zone.

Dimensi	Deskripsi
`FilterId`	Dimensi ini memfilter konfigurasi metrik yang Anda tentukan untuk metrik permintaan di bucket. Saat Anda membuat konfigurasi metrik, Anda menentukan ID filter (misalnya, prefiks, tag, atau titik akses). Untuk informasi lebih lanjut, lihat Membuat konfigurasi metrik.

Dimensi replikasi S3 di CloudWatch

Dimensi berikut digunakan untuk menyaring metrik Replikasi S3.

Dimensi	Deskripsi
`SourceBucket`	Nama objek ember direplikasi dari.
`DestinationBucket`	Nama objek bucket direplikasi ke.
`RuleId`	Pengidentifikasi unik untuk aturan yang memicu metrik replikasi ini diperbarui.

Dimensi Lensa Penyimpanan S3 di CloudWatch

Untuk daftar dimensi yang digunakan untuk memfilter metrik Lensa Penyimpanan S3 CloudWatch, lihat. Dimensi

Dimensi permintaan Lambda Objek S3 di CloudWatch

Dimensi berikut digunakan untuk memfilter data dari Titik Akses Lambda Objek.

Dimensi	Deskripsi
`AccessPointName`	Nama titik akses tempat permintaan sedang dibuat.
`DataSourceARN`	Sumber Titik Akses Lambda Objek mengambil data dari. Jika permintaan memanggil fungsi Lambda, ini merujuk ke Nama Sumber Daya Amazon Lambda (ARN). Kalau tidak, ini mengacu pada titik akses ARN.

Mengakses metrik CloudWatch

Anda dapat menggunakan prosedur berikut untuk melihat metrik penyimpanan untuk Amazon S3. Untuk melibatkan metrik Amazon S3, Anda harus mengatur stempel waktu mulai dan selesai. Untuk metrik untuk periode 24 jam tertentu, atur periode waktu menjadi 86400 detik, jumlah detik dalam sehari. Selain itu, ingatlah untuk mengatur dimensi `BucketName` dan `StorageType`.

Menggunakan AWS CLI

Misalnya, jika Anda ingin menggunakan file AWS CLI untuk mendapatkan rata-rata ukuran bucket tertentu dalam byte, Anda dapat menggunakan perintah berikut:

```
aws cloudwatch get-metric-statistics --metric-name BucketSizeBytes --namespace AWS/S3
 --start-time 2016-10-19T00:00:00Z --end-time 2016-10-20T00:00:00Z --statistics Average
 --unit Bytes --region us-west-2 --dimensions Name=BucketName,Value=DOC-EXAMPLE-BUCKET
 Name=StorageType,Value=StandardStorage --period 86400 --output json
```

Contoh ini menghasilkan hasil berikut.

```
{
    "Datapoints": [
        {
            "Timestamp": "2016-10-19T00:00:00Z",
            "Average": 1025328.0,
            "Unit": "Bytes"
        }
    ],
    "Label": "BucketSizeBytes"
}
```

Menggunakan konsol S3

Untuk melihat metrik menggunakan konsol Amazon CloudWatch

1. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Di panel navigasi, pilih Metrik.
3. Pilih namespace S3.
4. (Opsional) Untuk melihat metrik, masukkan nama metrik dalam kotak pencarian.
5. (Opsional) Untuk memfilter berdasarkan StorageTypedimensi, masukkan nama kelas penyimpanan di kotak pencarian.

Untuk melihat daftar metrik valid yang disimpan untuk Anda Akun AWS dengan menggunakan AWS CLI

- Pada jendela perintah, gunakan perintah berikut.

```
aws cloudwatch list-metrics --namespace "AWS/S3"
```

Untuk informasi selengkapnya tentang izin yang diperlukan untuk mengakses CloudWatch dasbor, lihat Izin CloudWatch dasbor Amazon di Panduan Pengguna Amazon CloudWatch .

CloudWatch konfigurasi metrik

Dengan metrik CloudWatch permintaan Amazon untuk Amazon S3, Anda dapat menerima metrik CloudWatch 1 menit, CloudWatch menyetel alarm, dan CloudWatch mengakses dasbor untuk near-real-time melihat operasi dan kinerja penyimpanan Amazon S3 Anda. Untuk aplikasi yang bergantung pada penyimpanan cloud, metrik ini memungkinkan Anda dengan cepat mengidentifikasi dan menindaklanjuti masalah operasional. Saat diaktifkan, metrik 1 menit ini tersedia di tingkat bucket Amazon S3, secara default.

Jika ingin mendapatkan metrik CloudWatch permintaan untuk objek dalam bucket, Anda harus membuat konfigurasi metrik untuk bucket. Untuk informasi selengkapnya, lihat Membuat konfigurasi CloudWatch metrik untuk semua objek di bucket.

Anda juga dapat menggunakan prefiks bersama, tag objek, atau titik akses untuk menentukan filter untuk metrik yang dikumpulkan. Metode mendefinisikan filter ini memungkinkan Anda menyelaraskan filter metrik ke aplikasi bisnis, alur kerja, atau organisasi internal tertentu. Untuk informasi selengkapnya, lihat Membuat konfigurasi metrik yang melakukan filter berdasarkan prefiks, tag objek, atau titik akses. Untuk informasi selengkapnya tentang CloudWatch metrik yang tersedia dan perbedaan antara metrik penyimpanan dan permintaan, lihat. Memantau metrik dengan Amazon CloudWatch

Ingat hal berikut ini saat menggunakan konfigurasi metrik:

- Anda dapat memiliki maksimum 1.000 konfigurasi metrik per bucket.
- Anda dapat memilih objek mana di dalam bucket untuk disertakan dalam konfigurasi metrik dengan menggunakan filter. Anda dapat memfilter pada prefiks bersama, tag objek, atau titik akses untuk menyelaraskan filter metrik ke aplikasi bisnis, alur kerja, atau organisasi internal tertentu. Untuk meminta metrik untuk seluruh bucket, buat konfigurasi metrik tanpa filter.

- Konfigurasi metrik hanya diperlukan untuk mengaktifkan metrik permintaan. Metrik penyimpanan harian tingkat bucket selalu diaktifkan, dan disediakan tanpa biaya tambahan. Saat ini, tidak mungkin mendapatkan metrik penyimpanan harian untuk subset objek yang difilter.
- Setiap konfigurasi metrik memungkinkan serangkaian penuh metrik permintaan yang tersedia. Metrik spesifik operasi (seperti `PostRequests`) dilaporkan hanya jika ada permintaan jenis tersebut untuk bucket atau filter Anda.
- Metrik permintaan dilaporkan untuk operasi tingkat objek. Juga dilaporkan untuk operasi yang membuat daftar konten bucket, seperti GET Bucket (List Objects), GET Bucket Object Versions, dan List Multipart Uploads, tetapi tidak dilaporkan untuk operasi lain pada bucket.
- Minta pemfilteran yang mendukung metrik berdasarkan prefiks, tag objek, atau titik akses, tetapi metrik penyimpanan tidak demikian.

Pengiriman CloudWatch metrik upaya terbaik

CloudWatch metrik disampaikan atas dasar upaya terbaik. Sebagian besar permintaan untuk objek Amazon S3 yang memiliki metrik permintaan menghasilkan titik data yang dikirim. CloudWatch

Kelengkapan dan ketepatan waktu metrik tidak dijamin. Titik data untuk permintaan tertentu mungkin dikembalikan dengan stempel waktu yang lebih lambat daripada ketika permintaan tersebut sebenarnya diproses. Titik data selama satu menit mungkin tertunda sebelum tersedia CloudWatch, atau mungkin tidak dikirimkan sama sekali. CloudWatch metrik permintaan memberi Anda gambaran tentang sifat lalu lintas terhadap bucket Anda dalam waktu hampir nyata. Ini tidak dimaksudkan untuk menjadi pencatatan akuntasi yang lengkap atas semua permintaan.

Berdasarkan sifat upaya terbaik dari fitur ini, laporan yang tersedia di Dasbor Manajemen Penagihan & Biaya mungkin menyertakan satu atau lebih permintaan akses yang tidak muncul dalam metrik bucket.

Untuk informasi selengkapnya tentang bekerja dengan CloudWatch metrik di Amazon S3, lihat topik berikut.

Topik

- Membuat konfigurasi CloudWatch metrik untuk semua objek di bucket
- Membuat konfigurasi metrik yang melakukan filter berdasarkan prefiks, tag objek, atau titik akses
- Menghapus filter metrik

Membuat konfigurasi CloudWatch metrik untuk semua objek di bucket

Saat mengonfigurasi metrik permintaan, Anda dapat membuat konfigurasi CloudWatch metrik untuk semua objek di bucket, atau memfilter berdasarkan awalan, tag objek, atau titik akses. Prosedur dalam topik ini menunjukkan kepada Anda cara membuat konfigurasi untuk semua objek dalam bucket Anda. Untuk membuat konfigurasi yang melakukan filter berdasarkan tag objek, prefiks, atau titik akses, lihat Membuat konfigurasi metrik yang melakukan filter berdasarkan prefiks, tag objek, atau titik akses.

Ada tiga jenis metrik Amazon untuk Amazon S3: CloudWatch metrik penyimpanan, metrik permintaan, dan metrik replikasi. Metrik penyimpanan dilaporkan satu kali sehari, dan diberikan kepada semua pelanggan tanpa biaya tambahan. Metrik permintaan tersedia dalam interval satu menit setelah beberapa latensi untuk diproses. Metrik permintaan ditagih dengan tarif standar CloudWatch . Anda harus memilih untuk meminta metrik dengan mengonfigurasinya di dalam konsol, atau menggunakan API Amazon S3. Metrik Replikasi S3 memberikan metrik terperinci untuk aturan replikasi dalam konfigurasi replikasi Anda. Dengan metrik replikasi, Anda dapat memantau minute-by-minute kemajuan dengan melacak byte yang tertunda, operasi tertunda, operasi yang gagal replikasi, dan latensi replikasi.

Untuk informasi selengkapnya tentang CloudWatch metrik untuk Amazon S3, lihat. Memantau metrik dengan Amazon CloudWatch

Anda dapat menambahkan konfigurasi metrik ke bucket melalui konsol Amazon S3, dengan AWS Command Line Interface (AWS CLI), atau dengan API REST Amazon S3.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang berisi objek yang Anda inginkan meminta metrik.
3. Pilih tab Metrik.
4. Di bagian bawah Metrik bucket, pilih Lihat bagan tambahan.
5. Pilih tab Minta metrik.
6. Pilih Buat filter.
7. Di dalam kotak Nama filter, masukkan nama filter Anda.

   Nama hanya dapat berisi huruf, angka, periode, tanda hubung, dan garis bawah. Kami sarankan menggunakan nama `EntireBucket` untuk filter yang berlaku di semua objek.

8. Di bagian bawah Cakupan filter, pilih Filter ini berlaku untuk semua objek dalam bucket.

   Anda juga dapat menentukan filter sehingga metrik hanya dikumpulkan dan dilaporkan pada subset objek dalam bucket. Untuk informasi selengkapnya, lihat Membuat konfigurasi metrik yang melakukan filter berdasarkan prefiks, tag objek, atau titik akses.
9. Pilih Simpan perubahan.
10. Pada tab Minta metrik, di bagian bawah Filter, pilih filter yang baru saja Anda buat.

    Setelah sekitar 15 menit, CloudWatch mulai melacak metrik permintaan ini. Anda dapat melihatnya di tab Minta metrik. Anda dapat melihat grafik untuk metrik di Amazon CloudWatch S3 atau konsol. Metrik permintaan ditagih dengan tarif standar CloudWatch . Untuk informasi selengkapnya, lihat CloudWatch harga Amazon.

Penggunaan API REST

Anda juga dapat menambahkan konfigurasi metrik secara terprogram dengan API REST Amazon S3. Untuk informasi selengkapnya tentang menambahkan dan bekerja dengan konfigurasi metrik, lihat topik berikut di Referensi API Amazon Simple Storage Service:

- Konfigurasi Metrik Bucket PUT
- Konfigurasi Metrik Bucket GET
- Daftar Konfigurasi Metrik Bucket
- Konfigurasi Metrik Bucket DELETE

Menggunakan AWS CLI

1. Instal dan atur AWS CLI. Untuk instruksi, lihat Menginstal, memperbarui, dan menghapus AWS CLI di AWS Command Line Interface Panduan Pengguna.
2. Buka terminal.
3. Jalankan perintah berikut untuk menambahkan konfigurasi metrik.

```
aws s3api put-bucket-metrics-configuration --endpoint https://s3.us-
west-2.amazonaws.com --bucket bucket-name --id metrics-config-id --metrics-
configuration '{"Id":"metrics-config-id"}'
```

Membuat konfigurasi metrik yang melakukan filter berdasarkan prefiks, tag objek, atau titik akses

Ada tiga jenis metrik Amazon untuk Amazon S3: CloudWatch metrik penyimpanan, metrik permintaan, dan metrik replikasi. Metrik penyimpanan dilaporkan satu kali sehari, dan diberikan kepada semua pelanggan tanpa biaya tambahan. Metrik permintaan tersedia dalam interval satu menit setelah beberapa latensi untuk diproses. Metrik permintaan ditagih dengan tarif standar CloudWatch . Anda harus memilih untuk meminta metrik dengan mengonfigurasinya di dalam konsol, atau menggunakan API Amazon S3. Metrik Replikasi S3 memberikan metrik terperinci untuk aturan replikasi dalam konfigurasi replikasi Anda. Dengan metrik replikasi, Anda dapat memantau minute-by-minute kemajuan dengan melacak byte yang tertunda, operasi tertunda, operasi yang gagal replikasi, dan latensi replikasi.

Untuk informasi selengkapnya tentang CloudWatch metrik untuk Amazon S3, lihat. Memantau metrik dengan Amazon CloudWatch

Saat mengonfigurasi CloudWatch metrik, Anda dapat membuat filter untuk semua objek di bucket, atau memfilter konfigurasi ke dalam grup objek terkait dalam satu bucket. Anda dapat memfilter objek dalam bucket untuk disertakan dalam konfigurasi metrik berdasarkan satu atau beberapa jenis filter berikut ini:

- Prefiks nama kunci objek–Meskipun model data Amazon S3 adalah struktur datar, Anda dapat menyimpulkan hierarki dengan menggunakan prefiks. Konsol Amazon S3 mendukung prefiks ini dengan konsep folder. Jika Anda memfilter menurut prefiks, objek yang memiliki prefiks yang sama akan disertakan dalam konfigurasi metrik. Untuk informasi selengkapnya tentang prefiks, lihat Organisasi objek menggunakan prefiks.
- Tag–Anda dapat menambahkan tag, yang merupakan pasangan nama nilai-kunci, untuk objek. Tag membantu Anda untuk menemukan dan menata objek dengan mudah. Anda juga dapat menggunakan tag sebagai filter untuk konfigurasi metrik. Untuk informasi selengkapnya tentang tag objek, lihat Mengategorikan penyimpanan Anda menggunakan tag.
- Titik akses–Titik Akses S3 diberi nama titik akhir jaringan yang dilampirkan ke bucket dan menyederhanakan pengelolaan akses data pada skala untuk set data bersama di S3. Saat Anda membuat filter titik akses, Amazon S3 menyertakan permintaan ke titik akses yang Anda tentukan dalam konfigurasi metrik. Untuk informasi selengkapnya, lihat Pemantauan dan pencatatan titik akses.

> **Note**
>
> Saat Anda membuat konfigurasi metrik yang memfilter berdasarkan titik akses, Anda harus menggunakan titik akses Amazon Resource Name (ARN), bukan alias titik akses. Pastikan Anda menggunakan ARN untuk titik akses itu sendiri, bukan ARN untuk objek tertentu. Untuk informasi lebih lanjut tentang titik akses ARN, lihat Menggunakan titik akses.

Jika Anda menetapkan filter, hanya permintaan yang beroperasi pada objek tunggal yang dapat cocok dengan filter dan disertakan dalam metrik yang dilaporkan. Permintaan suka DeleteObjectsdan `ListObjects` permintaan tidak menampilkan metrik apa pun untuk konfigurasi dengan filter.

Untuk meminta pemfilteran yang lebih kompleks, pilih dua atau lebih elemen. Hanya objek yang memiliki semua elemen tersebut yang disertakan dalam konfigurasi metrik. Jika Anda tidak mengatur filter, semua objek dalam bucket disertakan dalam konfigurasi metrik.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang berisi objek yang Anda inginkan meminta metrik.
3. Pilih tab Metrik.
4. Di bagian bawah Metrik bucket, pilih Lihat bagan tambahan.
5. Pilih tab Minta metrik.
6. Pilih Buat filter.
7. Di dalam kotak Nama filter, masukkan nama filter Anda.

   Nama hanya dapat berisi huruf, angka, periode, tanda hubung, dan garis bawah.
8. Di bawah Cakupan filter, pilih Batasi cakupan filter ini menggunakan prefiks, tag objek, dan Titik Akses S3, atau kombinasi ketiganya.
9. Di bawah Jenis filter, pilih setidaknya satu jenis filter: Awalan, Tag objek, atau Titik akses.
10. Untuk menentukan filter prefiks dan membatasi cakupan filter ke satu jalur, di kotak Prefiks, masukkan prefiks.
11. Untuk menentukan filter tag objek, di bawah Tag objek, pilih Tambah tag, lalu masukkan tag Kunci dan Nilai.

12. Untuk menentukan filter titik akses, di bidang Titik Akses S3, masukkan titik akses ARN, atau pilih Telusuri S3 untuk menavigasi ke titik akses.

> **Important**
>
> Anda tidak dapat memasukkan alias titik akses. Anda harus memasukkan ARN untuk titik akses itu sendiri, bukan ARN untuk objek tertentu.

13. Pilih Simpan perubahan.

    Amazon S3 membuat filter yang menggunakan prefiks, tag, atau titik akses yang Anda tentukan.

14. Pada tab Minta metrik, di bagian bawah Filter, pilih filter yang baru saja Anda buat.

    Sekarang Anda telah membuat filter yang membatasi cakupan metrik permintaan berdasarkan prefiks, tag objek, atau titik akses. Sekitar 15 menit setelah CloudWatch mulai melacak metrik permintaan ini, Anda dapat melihat bagan untuk metrik di Amazon CloudWatch S3 dan konsol. Metrik permintaan ditagih dengan tarif standar CloudWatch . Untuk informasi selengkapnya, lihat CloudWatch harga Amazon.

    Anda juga dapat mengonfigurasi metrik permintaan di tingkat bucket. Untuk informasi, lihat Membuat konfigurasi CloudWatch metrik untuk semua objek di bucket.

Menggunakan AWS CLI

1. Instal dan atur AWS CLI. Untuk instruksi, lihat Menginstal, memperbarui, dan menghapus AWS CLI di AWS Command Line Interface Panduan Pengguna.
2. Buka terminal.
3. Untuk menambahkan konfigurasi metrik, jalankan salah satu dari perintah berikut ini:

   Example : Untuk memfilter berdasarkan prefiks

```
aws s3api put-bucket-metrics-configuration --bucket DOC-EXAMPLE-BUCKET1 --
id metrics-config-id --metrics-configuration '{"Id":"metrics-config-id", "Filter":
{"Prefix":"prefix1"}} '
```

Example : Untuk memfilter berdasarkan tag

```
aws s3api put-bucket-metrics-configuration --bucket DOC-EXAMPLE-BUCKET1 --
id metrics-config-id --metrics-configuration '{"Id":"metrics-config-id", "Filter":
{"Tag": {"Key": "string", "Value": "string"}} '
```

Example : Untuk memfilter berdasarkan titik akses

```
aws s3api put-bucket-metrics-configuration --bucket DOC-EXAMPLE-BUCKET1 --
id metrics-config-id --metrics-configuration '{"Id":"metrics-config-id", "Filter":
{"AccessPointArn":"arn:aws:s3:Region:account-id:accesspoint/access-point-name"}} '
```

Example : Untuk memfilter berdasarkan prefiks, tag, dan titik akses

```
aws s3api put-bucket-metrics-configuration --endpoint https://
s3.Region.amazonaws.com --bucket DOC-EXAMPLE-BUCKET1 --id metrics-config-id --
metrics-configuration '
{
    "Id": "metrics-config-id",
    "Filter": {
        "And": {
            "Prefix": "string",
            "Tags": [
                {
                    "Key": "string",
                    "Value": "string"
                }
            ],
            "AccessPointArn": "arn:aws:s3:Region:account-id:accesspoint/access-
point-name"
        }
    }
}'
```

Penggunaan API REST

Anda juga dapat menambahkan konfigurasi metrik secara terprogram dengan API REST Amazon S3. Untuk informasi selengkapnya tentang menambahkan dan bekerja dengan konfigurasi metrik, lihat topik berikut di Referensi API Amazon Simple Storage Service:

- Konfigurasi Metrik Bucket PUT
- Konfigurasi Metrik Bucket GET
- Daftar Konfigurasi Metrik Bucket
- Konfigurasi Metrik Bucket DELETE

Menghapus filter metrik

Anda dapat menghapus filter metrik CloudWatch permintaan Amazon jika Anda tidak lagi membutuhkannya. Saat menghapus filter, Anda tidak lagi dikenakan biaya untuk metrik permintaan yang menggunakan filter tersebut filter khusus. Namun, Anda akan terus dibebankan untuk konfigurasi filter lainnya yang ada.

Saat menghapus filter, Anda tidak lagi dapat menggunakan filter untuk metrik permintaan. Tindakan menghapus filter tidak dapat diurungkan.

Untuk informasi selengkapnya tentang membuat filter metrik permintaan, lihat topik-topik berikut:

- Membuat konfigurasi CloudWatch metrik untuk semua objek di bucket
- Membuat konfigurasi metrik yang melakukan filter berdasarkan prefiks, tag objek, atau titik akses

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket Anda.
3. Pilih tab Metrik.
4. Di bagian bawah Metrik bucket, pilih Lihat bagan tambahan.
5. Pilih tab Minta metrik.
6. Pilih Kelola filter.
7. Pilih filter Anda.

   > **Important**
   >
   > Tindakan menghapus filter tidak dapat diurungkan.

8. Pilih Hapus.

Amazon S3 menghapus filter Anda.

Penggunaan API REST

Anda juga dapat menambahkan konfigurasi metrik secara terprogram dengan API REST Amazon S3. Untuk informasi selengkapnya tentang menambahkan dan bekerja dengan konfigurasi metrik, lihat topik berikut di Referensi API Amazon Simple Storage Service:

- Konfigurasi Metrik Bucket PUT
- Konfigurasi Metrik Bucket GET
- Daftar Konfigurasi Metrik Bucket
- Konfigurasi Metrik Bucket DELETE

Notifikasi Peristiwa Amazon S3

Anda dapat menggunakan fitur Notifikasi Peristiwa Amazon S3 untuk menerima pemberitahuan ketika peristiwa tertentu terjadi di bucket S3 Anda. Untuk mengaktifkan notifikasi, tambahkan konfigurasi notifikasi yang mengidentifikasi peristiwa yang ingin dipublikasikan Amazon S3. Pastikan itu juga mengidentifikasi tujuan tempat Anda ingin Amazon S3 mengirim notifikasi. Anda menyimpan konfigurasi ini di sub-sumber daya notifikasi yang terkait dengan bucket. Untuk informasi selengkapnya, lihat Opsi konfigurasi bucket. Amazon S3 menyediakan API bagi Anda untuk dapat mengelola sub-sumber daya ini.

> **Important**
>
> Pemberitahuan peristiwa Amazon S3 dirancang untuk dikirimkan setidaknya satu kali. Biasanya, pemberitahuan peristiwa dikirimkan dalam hitungan detik, tetapi terkadang perlu waktu satu menit atau lebih.

Gambaran Umum Notifikasi Peristiwa Amazon S3

Saat ini, Amazon S3 dapat menerbitkan pemberitahuan untuk peristiwa berikut:

- Objek baru yang dibuat peristiwa
- Peristiwa penghapusan objek

- Kembalikan peristiwa objek
- Mengurangi objek Penyimpanan Redundansi (RRS) yang hilang
- Peristiwa replikasi
- Acara kedaluwarsa Siklus Hidup S3
- Peristiwa transisi Siklus Hidup S3
- S3 Intelligent-Tiering peristiwa arsip otomatis
- Peristiwa pemberian tag objek
- Peristiwa objek ACL PUT

Untuk deskripsi lengkap semua jenis peristiwa yang didukung, lihat Jenis event yang didukung untuk SQS, SNS, dan Lambda.

Amazon S3 dapat mengirimkan pesan pemberitahuan peristiwa ke tujuan berikut. Anda menentukan nilai Amazon Resource Name (ARN) dari tujuan ini dalam konfigurasi notifikasi.

- Topik Amazon Simple Notification Service (Amazon SNS)
- Antrean Amazon Simple Queue Service (Amazon SQS)
- AWS Lambda fungsi
- Amazon EventBridge

Untuk informasi selengkapnya, lihat Tujuan peristiwa yang didukung.

> **Note**
>
> Antrean Amazon Simple Queue Service FIFO (First-In-First-Out) tidak didukung sebagai tujuan pemberitahuan peristiwa Amazon S3. Untuk mengirim pemberitahuan untuk acara Amazon S3 ke antrian FIFO Amazon SQS, Anda dapat menggunakan Amazon. EventBridge Untuk informasi selengkapnya, lihat Mengaktifkan Amazon EventBridge.

> **Warning**
>
> Jika notifikasi Anda akhirnya menulis ke bucket yang sama yang memicu notifikasi, ini dapat menyebabkan loop eksekusi. Misalnya, jika bucket memicu fungsi Lambda setiap kali suatu

objek diunggah, dan fungsinya mengunggah sebuah objek ke bucket, maka fungsi tersebut secara tidak langsung memicu fungsi itu sendiri. Untuk menghindari hal ini, gunakan dua bucket, atau konfigurasikan pemicu agar hanya berlaku pada prefiks yang digunakan untuk objek masuk.
Untuk informasi selengkapnya dan contoh penggunaan notifikasi Amazon S3 dengan AWS Lambda, lihat Menggunakan AWS Lambda dengan Amazon S3 di AWS Lambda Panduan Pengembang.

Untuk informasi selengkapnya tentang jumlah konfigurasi notifikasi peristiwa yang dapat Anda buat per bucket, lihat Kuota layanan Amazon S3 dalam AWS Referensi Umum.

Untuk informasi selengkapnya tentang notifikasi Peristiwa, lihat bagian berikut.

Topik

- Jenis dan tujuan pemberitahuan peristiwa
- Menggunakan Amazon SQS, Amazon SNS, dan Lambda
- Menggunakan EventBridge

Jenis dan tujuan pemberitahuan peristiwa

Amazon S3 mendukung beberapa jenis tujuan notifikasi peristiwa tempat notifikasi dapat diterbitkan. Anda dapat menentukan jenis peristiwa dan tujuan ketika mengonfigurasi notifikasi peristiwa Anda. Hanya satu tujuan yang dapat ditentukan untuk setiap pemberitahuan peristiwa. Pemberitahuan peristiwa Amazon S3 mengirim satu entri peristiwa untuk setiap pesan notifikasi.

Topik

- Tujuan peristiwa yang didukung
- Jenis event yang didukung untuk SQS, SNS, dan Lambda
- Jenis acara yang didukung untuk Amazon EventBridge
- Pemesanan acara dan duplikat acara

Tujuan peristiwa yang didukung

Amazon S3 dapat mengirimkan pesan pemberitahuan peristiwa ke tujuan berikut.

- Topik Amazon Simple Notification Service (Amazon SNS)
- Antrean Amazon Simple Queue Service (Amazon SQS)
- AWS Lambda
- Amazon EventBridge

Namun, hanya satu jenis tujuan yang dapat ditentukan untuk setiap pemberitahuan peristiwa.

> **Note**
>
> Anda harus memberikan izin Amazon S3 untuk mem-posting pesan ke topik Amazon SNS atau antrean Amazon SQS. Anda juga harus memberikan izin Amazon S3 untuk menjalankan AWS Lambda fungsi atas nama Anda. Untuk petunjuk tentang cara memberikan izin ini, lihat memberikan izin untuk memublikasikan pesan pemberitahuan peristiwa ke tujuan.

Topik Amazon SNS

Amazon SNS adalah layanan perpesanan push yang fleksibel, dan dikelola penuh. Anda dapat menggunakan layanan ini untuk mendorong pesan ke perangkat seluler atau layanan terdistribusi. Dengan SNS, Anda dapat memublikasikan pesan satu kali, dan mengirimkannya satu atau beberapa kali. Saat ini SNS Standar hanya diizinkan sebagai tujuan notifikasi peristiwa S3, sedangkan SNS FIFO tidak diizinkan.

Amazon SNS mengoordinasikan dan mengelola pengiriman dan pengiriman pesan ke titik akhir atau klien yang berlangganan. Anda dapat menggunakan konsol Amazon SNS untuk membuat topik Amazon SNS yang dapat dikirimkan pemberitahuan Anda.

Topiknya harus Wilayah AWS sama dengan bucket Amazon S3 Anda. Untuk petunjuk tentang cara membuat topik Amazon SNS, lihat Memulai dengan Amazon SNS di Panduan Pengembang Amazon Simple Notification Service dan FAQ Amazon SNS.

Sebelum Anda dapat menggunakan topik Amazon SNS yang Anda buat sebagai tujuan pemberitahuan peristiwa, Anda memerlukan hal berikut:

- Amazon Resource Name (ARN) untuk topik Amazon SNS
- Berlangganan topik Amazon SNS yang valid. Dengan itu, pelanggan topik diberi tahu saat pesan dipublikasikan ke topik Amazon SNS Anda.

Antrean Amazon SQS

Amazon SQS menawarkan antrean hostingan yang andal dan dapat diskalakan untuk menyimpan pesan saat mereka bepergian antar komputer. Anda dapat menggunakan Amazon SQS untuk mengirimkan volume data apa pun tanpa memerlukan layanan lain untuk selalu tersedia. Anda dapat menggunakan konsol Amazon SQS untuk membuat antrean Amazon SQS yang dapat dikirimkan pemberitahuan Anda.

Antrian Amazon SQS harus sama Wilayah AWS dengan bucket Amazon S3 Anda. Untuk petunjuk tentang cara membuat antrean Amazon SQS, lihat Apa itu Amazon Simple Queue Service dan Memulai dengan Amazon SQS di Panduan Pengembang Amazon Simple Storage Service.

Sebelum Anda dapat menggunakan antrean Amazon SQS sebagai tujuan pemberitahuan peristiwa, Anda memerlukan berikut ini:

- Amazon Resource Name (ARN) untuk antrean Amazon SQS

> **Note**
>
> Antrean Amazon Simple Queue Service FIFO (First-In-First-Out) tidak didukung sebagai tujuan pemberitahuan peristiwa Amazon S3. Untuk mengirim pemberitahuan untuk acara Amazon S3 ke antrian FIFO Amazon SQS, Anda dapat menggunakan Amazon. EventBridge Untuk informasi selengkapnya, lihat Mengaktifkan Amazon EventBridge.

Fungsi Lambda

Anda dapat menggunakan AWS Lambda untuk memperluas AWS layanan lain dengan logika khusus, atau membuat backend Anda sendiri yang beroperasi pada AWS skala, kinerja, dan keamanan. Dengan Lambda, Anda dapat membuat aplikasi terpisah berbasis peristiwa yang berjalan hanya saat diperlukan. Anda juga dapat menggunakannya untuk menskalakan aplikasi ini secara otomatis dari beberapa permintaan sehari hingga ribuan detik.

Lambda dapat menjalankan kode kustom sebagai respons terhadap peristiwa bucket Amazon S3. Anda unggah kode kustom Anda ke Lambda dan membuat apa yang disebut fungsi Lambda. Saat Amazon S3 mendeteksi peristiwa dari jenis tertentu, Amazon S3 dapat memublikasikan acara ke AWS Lambda dan memanggil fungsi Anda di Lambda. Sebagai respons, Lambda menjalankan fungsi Anda. Salah satu jenis peristiwa yang mungkin dideteksi, misalnya, adalah peristiwa yang dibuat objek.

Anda dapat menggunakan AWS Lambda konsol untuk membuat fungsi Lambda yang menggunakan AWS infrastruktur untuk menjalankan kode atas nama Anda. Fungsi Lambda harus berada di Wilayah yang sama dengan bucket S3. Anda juga harus memiliki nama atau ARN dari fungsi Lambda untuk mengatur fungsi Lambda sebagai tujuan notifikasi peristiwa.

> **Warning**
>
> Jika notifikasi Anda akhirnya menulis ke bucket yang sama yang memicu notifikasi, ini dapat menyebabkan loop eksekusi. Misalnya, jika bucket memicu fungsi Lambda setiap kali suatu objek diunggah, dan fungsinya mengunggah sebuah objek ke bucket, maka fungsi tersebut secara tidak langsung memicu fungsi itu sendiri. Untuk menghindari hal ini, gunakan dua bucket, atau konfigurasikan pemicu agar hanya berlaku pada prefiks yang digunakan untuk objek masuk.
>
> Untuk informasi selengkapnya dan contoh penggunaan notifikasi Amazon S3 dengan AWS Lambda, lihat Menggunakan AWS Lambda dengan Amazon S3 di AWS Lambda Panduan Pengembang.

Amazon EventBridge

Amazon EventBridge adalah bus acara tanpa server, yang menerima acara dari AWS layanan. Anda dapat mengatur aturan untuk mencocokkan peristiwa dan mengirimkannya ke target, seperti layanan AWS atau titik akhir HTTP. Untuk informasi selengkapnya, lihat Apa yang ada EventBridge di Panduan EventBridge Pengguna Amazon.

Tidak seperti tujuan lain, Anda dapat mengaktifkan atau menonaktifkan acara yang akan dikirimkan EventBridge untuk ember. Jika Anda mengaktifkan pengiriman, semua acara dikirim ke EventBridge. Selain itu, Anda dapat menggunakan EventBridge aturan untuk merutekan acara ke target tambahan.

Jenis event yang didukung untuk SQS, SNS, dan Lambda

Amazon S3 dapat menerbitkan peristiwa dari jenis berikut. Anda menentukan jenis peristiwa ini di konfigurasi pemberitahuan.

Jenis peristiwa	Deskripsi
s3: TestEvent	Ketika notifikasi diaktifkan, Amazon S3 menerbitkan notifikasi tes. Hal ini untuk memastikan bahwa topik ada

Jenis peristiwa	Deskripsi
	dan bahwa pemilik bucket memiliki izin untuk mempublik asikan topik tertentu. Jika mengaktifkan notifikasi gagal, Anda tidak menerima notifikasi tes.
s3:ObjectCreated: * s3 ::Masukan ObjectCreated s3 ::Posting ObjectCreated s3 ::Salin ObjectCreated s3:: ObjectCreated CompleteM ultipartUpload	Operasi API Amazon S3 seperti PUT, POST, dan COPY dapat membuat objek. Dengan jenis peristiwa ini, Anda dapat mengaktifkan notifikasi saat objek dibuat menggunak an operasi API tertentu. Atau, Anda dapat menggunakan jenis peristiwa `s3:ObjectCreated:*` untuk meminta notifikasi terlepas dari API yang digunakan untuk membuat objek. `s3:ObjectCreated:CompleteMultipartUp load` termasuk objek yang dibuat menggunakan UploadPartCopyuntuk operasi Salin.

Jenis peristiwa	Deskripsi
s3:ObjectRemoved: * s3 ::Hapus ObjectRemoved s3:: ObjectRemoved DeleteMarkerCreated	Dengan menggunakan jenis ObjectRemovedacara, Anda dapat mengaktifkan notifikasi saat objek atau kumpulan objek dihapus dari ember. Anda dapat meminta pemberitahuan saat objek dihapus atau objek versi dihapus secara permanen menggunakan jenis peristiwa `s3:ObjectRemoved:Delete` . Atau, Anda dapat meminta notifikasi saat penanda penghapusan dibuat untuk objek versi menggunakan `s3:ObjectRemoved:DeleteMarkerCreated` . Untuk petunjuk cara menghapus objek berversi, lihat Menghapus versi objek dari bucket dengan dukungan Penentuan Versi. Anda juga dapat menggunakan wildcard `s3:ObjectRemoved:*` untuk meminta pemberitahuan kapan pun sebuah objek dihapus. Pemberitahuan peristiwa ini tidak memperingatkan Anda untuk penghapusan otomatis dari konfigurasi siklus aktif atau dari operasi yang gagal.
s3:ObjectRestore: * s3 ::Posting ObjectRestore s3 ::Selesai ObjectRestore s3 ::Hapus ObjectRestore	Dengan menggunakan jenis ObjectRestoreacara, Anda dapat menerima pemberitahuan untuk inisiasi dan penyelesaian acara saat memulihkan objek dari kelas penyimpanan Pengambilan Fleksibel Gletser S3, kelas penyimpanan S3 Glacier Deep Archive, tingkat Akses Arsip Tingkat Cerdas S3, dan tingkat Akses Arsip Tingkat Cerdas S3. Anda juga dapat menerima pemberitahuan ketika salinan objek yang dipulihkan kedaluwarsa. Jenis peristiwa `s3:ObjectRestore:Post` memberi tahu Anda tentang inisiasi restorasi objek. Jenis peristiwa `s3:ObjectRestore:Completed` memberi tahu Anda tentang penyelesaian restorasi. Jenis peristiwa `s3:ObjectRestore:Delete` memberi tahu Anda saat salinan sementara objek yang dipulihkan kedaluwarsa.

Jenis peristiwa	Deskripsi
s3: ReducedRedundancyLostObject	Anda menerima peristiwa notifikasi ini saat Amazon S3 mendeteksi bahwa objek kelas penyimpanan RRS hilang.
s3:Replication:* S3: replikasi: OperationFailedReplication S3: replikasi: OperationMissedThreshold S3: replikasi: OperationReplicatedAfterThreshold S3: replikasi: OperationNotTracked	Dengan menggunakan jenis peristiwa Replikasi, Anda dapat menerima notifikasi untuk konfigurasi yang memiliki metrik Replikasi S3 atau Kontrol Waktu Replikasi S3 (S3 RTC) aktif. Anda dapat memantau minute-by-minute kemajuan peristiwa replikasi dengan melacak byte yang tertunda, operasi tertunda, dan latensi replikasi. Untuk informasi tentang metrik replikasi, lihat Memantau progres dengan metrik replikasi dan Notifikasi Peristiwa S3. Jenis peristiwa `s3:Replication:OperationFailedReplication` memberi tahu Anda saat objek yang memenuhi syarat untuk replikasi gagal mereplikasi. Jenis peristiwa `s3:Replication:OperationMissedThreshold` memberi tahu Anda saat objek yang memenuhi syarat untuk replikasi melebihi ambang batas 15 menit untuk replikasi. Jenis peristiwa `s3:Replication:OperationReplicatedAfterThreshold` memberi tahu Anda saat objek yang memenuhi syarat untuk replikasi yang menggunakan Kontrol Waktu Replikasi S3 bereplikasi setelah ambang batas 15 menit. Jenis peristiwa `s3:Replication:OperationNotTracked` memberi tahu Anda saat objek yang memenuhi syarat untuk replikasi yang menggunakan Kontrol Waktu Replikasi S3 tetapi tidak lagi dilacak dengan metrik replikasi.

Jenis peristiwa	Deskripsi
s3:LifecycleExpiration: * s3 ::Hapus LifecycleExpiration s3:: LifecycleExpiration DeleteMarkerCreated	Dengan menggunakan jenis LifecycleExpirationacara, Anda dapat menerima pemberitahuan saat Amazon S3 menghapus objek berdasarkan konfigurasi Siklus Hidup S3 Anda. Jenis peristiwa `s3:LifecycleExpiration:Delete` akan memberi tahu Anda saat objek dalam bucket yang tidak berversi dihapus. Ini juga memberi tahu Anda ketika versi objek dihapus secara permanen oleh konfigurasi Siklus Hidup S3. Jenis peristiwa `s3:LifecycleExpiration:DeleteMarkerCreated` akan memberi tahu Anda saat Siklus Hidup S3 membuat penanda hapus saat versi objek saat ini dalam bucket berversi dihapus.
s3: LifecycleTransition	Anda menerima peristiwa notifikasi ini saat objek dialihkan ke kelas penyimpanan Amazon S3 lainnya dengan konfigurasi Siklus Hidup S3.
s3: IntelligentTiering	Anda menerima peristiwa notifikasi ini ketika objek dalam kelas penyimpanan S3 Intelligent-Tiering pindah ke tingkat Archive Access atau tingkat Deep Archive Access.
s3:ObjectTagging: * s3 ::Masukan ObjectTagging s3 ::Hapus ObjectTagging	Dengan menggunakan jenis ObjectTaggingacara, Anda dapat mengaktifkan notifikasi saat tag objek ditambahkan atau dihapus dari objek. Jenis peristiwa `s3:ObjectTagging:Put` memberi tahu Anda saat tag PUT pada objek atau tag yang ada diperbarui. Jenis peristiwa `s3:ObjectTagging:Delete` memberi tahu Anda saat tag dihapus dari objek.
s3 ::Masukan ObjectAcl	Anda menerima peristiwa pemberitahuan ini ketika ACL PUT pada objek atau ketika ACL yang ada diubah. Peristiwa tidak dihasilkan ketika permintaannya tidak menghasilkan perubahan pada ACL objek.

Jenis acara yang didukung untuk Amazon EventBridge

Untuk daftar jenis acara yang akan dikirimkan Amazon S3 ke Amazon EventBridge, lihat. Menggunakan EventBridge

Pemesanan acara dan duplikat acara

Pemberitahuan Acara Amazon S3 dirancang untuk mengirimkan pemberitahuan setidaknya sekali, tetapi pemberitahuan tersebut tidak dijamin tiba dalam urutan yang sama dengan peristiwa yang terjadi. Pada kesempatan yang jarang terjadi, mekanisme coba ulang Amazon S3 dapat menyebabkan duplikat Pemberitahuan Acara S3 untuk peristiwa objek yang sama. Untuk selengkapnya tentang penanganan kejadian duplikat atau di luar pesanan, lihat Mengelola pemesanan acara dan duplikat peristiwa dengan Pemberitahuan Acara Amazon S3 di AWS Blog Penyimpanan.

Menggunakan Amazon SQS, Amazon SNS, dan Lambda

Mengaktifkan notifikasi adalah operasi tingkat bucket. Anda menyimpan konfigurasi notifikasi di sub-sumber daya notifikasi yang berasosiasi dengan bucket. Setelah Anda membuat atau mengubah konfigurasi notifikasi bucket, biasanya diperlukan waktu sekitar lima menit sampai perubahan diterapkan. `s3:TestEvent` terjadi ketika notifikasi pertama kali diaktifkan. Anda dapat menggunakan salah satu metode berikut untuk mengelola konfigurasi pemberitahuan:

- Menggunakan konsol Amazon S3—Anda dapat menggunakan UI konsol untuk mengatur konfigurasi notifikasi pada bucket tanpa harus tulis kode apa pun. Untuk informasi selengkapnya, lihat Mengaktifkan dan mengonfigurasi notifikasi peristiwa menggunakan konsol Amazon S3.
- Menggunakan AWS SDK secara terprogram — Secara internal, konsol dan SDK memanggil Amazon S3 REST API untuk mengelola subresource notifikasi yang terkait dengan bucket. Untuk contoh konfigurasi notifikasi yang menggunakan SDK AWS , lihat Panduan: Mengonfigurasi bucket untuk notifikasi (topik SNS atau antrean SQS).

> **Note**
>
> Anda juga dapat melakukan panggilan API REST Amazon S3 secara langsung dari kode Anda. Namun, ini dapat merepotkan karena untuk melakukannya Anda harus menulis kode untuk mengautentikasi permintaan Anda.

Terlepas dari metode yang Anda gunakan, Amazon S3 menyimpan konfigurasi notifikasi sebagai XL di sub-sumber daya notifikasi yang terkait dengan bucket. Untuk informasi tentang sub-sumber daya bucket, lihat Opsi konfigurasi bucket.

Topik

- memberikan izin untuk memublikasikan pesan pemberitahuan peristiwa ke tujuan
- Mengaktifkan dan mengonfigurasi notifikasi peristiwa menggunakan konsol Amazon S3
- Mengonfigurasi notifikasi peristiwa secara terprogram
- Panduan: Mengonfigurasi bucket untuk notifikasi (topik SNS atau antrean SQS)
- Mengonfigurasi notifikasi peristiwa dengan filter nama kunci objek
- Struktur pesan peristiwa

memberikan izin untuk memublikasikan pesan pemberitahuan peristiwa ke tujuan

Anda harus memberikan izin yang diperlukan kepada pengguna utama Amazon S3 untuk memanggil API terkait guna menerbitkan pesan ke topik SNS, antrean SQS, atau fungsi Lambda. Hal ini agar Amazon S3 dapat memublikasikan pesan pemberitahuan peristiwa ke tujuan.

Untuk memecahkan masalah memublikasikan pesan pemberitahuan peristiwa ke tujuan, lihat Memecahkan masalah untuk memublikasikan pemberitahuan peristiwa Amazon S3 ke topik Layanan Pemberitahuan Sederhana Amazon .

Topik

- Memberikan izin untuk menjalankan fungsi AWS Lambda
- memberikan izin untuk memublikasikan pesan ke topik SNS atau antrean SQS

Memberikan izin untuk menjalankan fungsi AWS Lambda

Amazon S3 menerbitkan pesan peristiwa AWS Lambda dengan menjalankan fungsi Lambda dan menyediakan pesan acara sebagai argumen.

Saat Anda menggunakan konsol Amazon S3 untuk mengonfigurasi notifikasi peristiwa pada bucket Amazon S3 untuk fungsi Lambda, konsol menyiapkan izin yang diperlukan pada fungsi Lambda. Ini agar Amazon S3 memiliki izin untuk menginvokasi fungsi dari bucket. Untuk informasi selengkapnya, lihat Mengaktifkan dan mengonfigurasi notifikasi peristiwa menggunakan konsol Amazon S3.

Anda juga dapat memberikan izin Amazon S3 dari AWS Lambda untuk menjalankan fungsi Lambda Anda. Untuk informasi selengkapnya, lihat Tutorial: Menggunakan AWS Lambda Amazon S3 di Panduan AWS Lambda Pengembang.

memberikan izin untuk memublikasikan pesan ke topik SNS atau antrean SQS

Untuk memberikan izin Amazon S3 untuk memublikasikan pesan ke topik SNS atau antrean SQS, lampirkan kebijakan AWS Identity and Access Management (IAM) ke topik SNS tujuan atau antrean SQS.

Sebagai contoh cara melampirkan kebijakan ke topik SNS atau antrean SQS, lihat Panduan: Mengonfigurasi bucket untuk notifikasi (topik SNS atau antrean SQS). Untuk informasi lebih lanjut tentang izin, lihat topik berikut:

- Contoh kasus untuk kendali akses Amazon SNS dalam Panduan Amazon Simple Notification Service
- Manajemen Identitas dan Akses di Amazon SQS dalam Panduan Pengembang Amazon Simple Queue Service

Kebijakan IAM untuk topik SNS tujuan

Berikut ini adalah contoh kebijakan AWS Identity and Access Management (IAM) yang Anda lampirkan ke topik SNS tujuan. Untuk petunjuk tentang cara menggunakan kebijakan ini untuk menyiapkan topik Amazon SNS tujuan untuk pemberitahuan peristiwa, lihat Panduan: Mengonfigurasi bucket untuk notifikasi (topik SNS atau antrean SQS).

```
{
    "Version": "2012-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "Example SNS topic policy",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "SNS:Publish"
            ],
            "Resource": "SNS-topic-ARN",
            "Condition": {
```

```
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:*:*:bucket-name"
                },
                "StringEquals": {
                    "aws:SourceAccount": "bucket-owner-account-id"
                }
            }
        }
    ]
}
```

Kebijakan IAM untuk antrean SQS tujuan

Berikut ini adalah contoh kebijakan IAM yang Anda lampirkan ke antrean SQS tujuan. Untuk petunjuk tentang cara menggunakan kebijakan ini untuk menyiapkan antrean Amazon SQS tujuan untuk pemberitahuan peristiwa, lihat Panduan: Mengonfigurasi bucket untuk notifikasi (topik SNS atau antrean SQS).

Untuk menggunakan kebijakan ini, Anda harus memperbarui ARN antrian Amazon SQS, nama bucket, dan ID pemilik bucket. Akun AWS

```
{
    "Version": "2012-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "example-statement-ID",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "SQS:SendMessage"
            ],
            "Resource": "arn:aws:sqs:Region:account-id:queue-name",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:*:*:awsexamplebucket1"
                },
                "StringEquals": {
                    "aws:SourceAccount": "bucket-owner-account-id"
                }
            }
```

```
        }
    ]
}
```

Untuk kebijakan Amazon SNS dan Amazon SQS IAM, Anda dapat menentukan `StringLike` kondisi dalam kebijakan, bukan kondisi `ArnLike`.

Ketika `ArnLike` digunakan, partisi, layanan, account-id, resource-type, dan sebagian resource-id bagian dari ARN harus memiliki pencocokan yang tepat dengan ARN dalam konteks permintaan. Hanya wilayah dan jalur sumber daya yang memungkinkan pencocokan sebagian.

Ketika `StringLike` digunakan sebagai pengganti `ArnLike`, pencocokan mengabaikan struktur ARN dan memungkinkan pencocokan paral, terlepas dari bagian yang diberi wildcard. Untuk informasi selengkapnya, lihat Elemen Kebijakan IAM JSON dalam Panduan Pengguna IAM.

```
"Condition": {
  "StringLike": { "aws:SourceArn": "arn:aws:s3:*:*:bucket-name" }
  }
```

AWS KMS kebijakan kunci

Jika antrean SQS atau topik SNS dienkripsi dengan kunci yang dikelola pelanggan AWS Key Management Service (AWS KMS), Anda harus memberikan izin utama layanan Amazon S3 untuk bekerja dengan topik atau antrian terenkripsi. Untuk memberikan izin pengguna utama layanan Amazon S3, tambahkan pernyataan berikut ke kebijakan kunci untuk CMK.

```
{
    "Version": "2012-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "example-statement-ID",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*"
```

```
        }
    ]
}
```

Untuk informasi selengkapnya tentang kebijakan AWS KMS utama, lihat Menggunakan kebijakan utama AWS KMS di Panduan AWS Key Management Service Pengembang.

Untuk informasi selengkapnya tentang penggunaan enkripsi sisi server dengan Amazon AWS KMS SQS dan Amazon SNS, lihat berikut ini:

- Pengelolaan kunci di Panduan Pengembang Layanan Notifikasi Sederhana Amazon.
- Pengelolaan kunci di Panduan Pengembang Layanan Antrean Sederhana Amazon.
- Mengenkripsi pesan yang diterbitkan untuk Amazon SNS dengan AWS KMS dalam Blog Komputasi AWS .

Mengaktifkan dan mengonfigurasi notifikasi peristiwa menggunakan konsol Amazon S3

Anda dapat mengaktifkan peristiwa bucket Amazon S3 tertentu untuk mengirimkan pesan notifikasi ke suatu destinasi kapan pun peristiwa tersebut terjadi. Bagian ini menjelaskan cara menggunakan konsol Amazon S3 untuk mengaktifkan pemberitahuan peristiwa. Untuk informasi tentang cara menggunakan notifikasi peristiwa dengan AWS SDK dan Amazon S3 REST API, lihat. Mengonfigurasi notifikasi peristiwa secara terprogram

Prasyarat: Sebelum dapat mengaktifkan notifikasi peristiwa untuk bucket Anda, Anda harus menyiapkan salah satu jenis tujuan dan kemudian mengonfigurasi izin. Untuk informasi lebih lanjut, lihat Tujuan peristiwa yang didukung dan memberikan izin untuk memublikasikan pesan pemberitahuan peristiwa ke tujuan.

> **Note**
>
> Antrean Amazon Simple Queue Service FIFO (First-In-First-Out) tidak didukung sebagai tujuan pemberitahuan peristiwa Amazon S3. Untuk mengirim pemberitahuan untuk acara Amazon S3 ke antrian FIFO Amazon SQS, Anda dapat menggunakan Amazon. EventBridge Untuk informasi selengkapnya, lihat Mengaktifkan Amazon EventBridge.

Topik

- Mengaktifkan notifikasi Amazon SNS, Amazon SQS, atau Lambda menggunakan konsol Amazon S3

Mengaktifkan notifikasi Amazon SNS, Amazon SQS, atau Lambda menggunakan konsol Amazon S3

Untuk mengaktifkan dan mengonfigurasi pemberitahuan peristiwa untuk bucket S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di Bucket pilih nama bucket yang ingin Anda aktifkan untuk peristiwa.
3. Pilih Properti.
4. Navigasikan ke Pemberitahuan Acara dan pilih Buat pemberitahuan peristiwa.
5. Di bagian Konfigurasi umum, sebutkan nama peristiwa deskriptif untuk pemberitahuan peristiwa Anda. Atau, Anda juga dapat menentukan prefiks dan sufiks untuk membatasi pemberitahuan ke objek dengan kunci yang berakhir dalam karakter yang ditentukan.

   a. Masukkan deskripsi untuk Nama peristiwa.

      Jika Anda tidak memasukkan nama, pengidentifikasi unik global (GUID) akan dibuat dan digunakan untuk nama.

   b. (Opsional) Untuk memfilter pemberitahuan peristiwa dengan prefiks, masukkan Prefiks.

      Misalnya, Anda dapat menyiapkan filter prefiks sehingga Anda menerima pemberitahuan hanya ketika file ditambahkan ke folder tertentu (misalnya, `images/`).

   c. (Opsional) Untuk memfilter pemberitahuan peristiwa dengan sufiks, masukkan Sufiks.

      Untuk informasi selengkapnya, lihat Mengonfigurasi notifikasi peristiwa dengan filter nama kunci objek.

6. Di bagian Jenis peristiwa, pilih satu atau beberapa jenis peristiwa yang ingin Anda terima pemberitahuannya.

   Untuk daftar jenis peristiwa berbeda, lihat Jenis event yang didukung untuk SQS, SNS, dan Lambda.

7. Di bagian Tujuan, pilih tujuan pemberitahuan peristiwa.

> **Note**
>
> Sebelum Anda dapat menerbitkan pemberitahuan peristiwa, Anda harus memberikan izin yang diperlukan kepada pengguna utama Amazon S3 untuk memanggil API terkait. Ini agar dapat menerbitkan pemberitahuan ke fungsi Lambda, topik SNS, atau antrean SQS.

a. Pilih jenis tujuan: Fungsi Lambda, Topik SNS, atau Antrean SQS.
b. Setelah memilih jenis tujuan Anda, pilih fungsi, topik, atau antrean dari daftar.
c. Atau, jika Anda ingin menentukan Amazon Resource Name (ARN), pilih Maemasukkan ARN dan memasukkan ARN.

Untuk informasi selengkapnya, lihat Tujuan peristiwa yang didukung.

8. Memilih Simpan perubahan, dan Amazon S3 mengirimkan pesan pengujian ke destinasi notifikasi peristiwa.

Mengonfigurasi notifikasi peristiwa secara terprogram

Secara default, notifikasi tidak diaktifkan untuk jenis peristiwa apa pun. Oleh karena itu, awalnya sub-sumber daya notifikasi menyimpan konfigurasi kosong.

```
<NotificationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
</NotificationConfiguration>
```

Untuk mengaktifkan pemberitahuan tentang peristiwa jenis tertentu, Anda mengganti XML dengan konfigurasi yang sesuai yang mengidentifikasi jenis peristiwa yang Anda ingin Amazon S3 untuk terbitkan dan tujuan di mana Anda ingin peristiwa tersebut diterbitkan. Untuk setiap tujuan, Anda menambahkan konfigurasi XML terkait.

Untuk mempublikasikan pesan peristiwa ke antrean SQS

Untuk mengatur antrean SQS sebagai tujuan notifikasi untuk satu atau beberapa jenis peristiwa, Anda menambahkan `QueueConfiguration`.

```
<NotificationConfiguration>
  <QueueConfiguration>
```

```
    <Id>optional-id-string</Id>
    <Queue>sqs-queue-arn</Queue>
    <Event>event-type</Event>
    <Event>event-type</Event>
     ...
  </QueueConfiguration>
   ...
</NotificationConfiguration>
```

Untuk memublikasikan pesan uji ke topik SNS

Untuk menetapkan topik SNS sebagai tujuan notifikasi untuk jenis peristiwa tertentu, Anda menambahkan `TopicConfiguration`.

```
<NotificationConfiguration>
  <TopicConfiguration>
     <Id>optional-id-string</Id>
     <Topic>sns-topic-arn</Topic>
     <Event>event-type</Event>
     <Event>event-type</Event>
      ...
  </TopicConfiguration>
   ...
</NotificationConfiguration>
```

Untuk memanggil AWS Lambda fungsi dan memberikan pesan acara sebagai argumen

Untuk mengatur fungsi Lambda sebagai tujuan notifikasi untuk jenis peristiwa tertentu, Anda menambahkan `CloudFunctionConfiguration`.

```
<NotificationConfiguration>
  <CloudFunctionConfiguration>
     <Id>optional-id-string</Id>
     <CloudFunction>cloud-function-arn</CloudFunction>
     <Event>event-type</Event>
     <Event>event-type</Event>
      ...
  </CloudFunctionConfiguration>
   ...
</NotificationConfiguration>
```

Untuk menghapus semua notifikasi yang dikonfigurasi pada bucket

Untuk menghapus semua notifikasi yang dikonfigurasi di bucket, simpan elemen `<NotificationConfiguration/>` kosong dalam sub-sumber daya notifikasi.

Saat Amazon S3 mendeteksi peristiwa dengan jenis tertentu, itu menerbitkan pesan dengan informasi peristiwa. Untuk informasi selengkapnya, lihat Struktur pesan peristiwa.

Untuk informasi lebih lanjut tentang mengonfigurasi notifikasi Peristiwa, lihat topik berikut:

- Panduan: Mengonfigurasi bucket untuk notifikasi (topik SNS atau antrean SQS).
- Mengonfigurasi notifikasi peristiwa dengan filter nama kunci objek

Panduan: Mengonfigurasi bucket untuk notifikasi (topik SNS atau antrean SQS)

Anda dapat menerima notifikasi Amazon S3 menggunakan Amazon Simple Notification Service (Amazon SNS); atau Amazon Simple Queue Service (Amazon SQS). Dalam panduan selanjutnya, Anda akan menambahkan konfigurasi notifikasi ke bucket Anda menggunakan topik Amazon SNS dan antrean Amazon SQS.

> **Note**
>
> Antrean Amazon Simple Queue Service FIFO (First-In-First-Out) tidak didukung sebagai tujuan pemberitahuan peristiwa Amazon S3. Untuk mengirim pemberitahuan untuk acara Amazon S3 ke antrian FIFO Amazon SQS, Anda dapat menggunakan Amazon. EventBridge Untuk informasi selengkapnya, lihat Mengaktifkan Amazon EventBridge.

Topik

- Ringkasan panduan
- Langkah 1: Buat antrean Amazon SQS
- Langkah 2: Buat Topik Amazon SNS
- Langkah 3: Menambahkan konfigurasi notifikasi ke bucket Anda
- Langkah 4: Menguji pengaturan

Ringkasan panduan

Panduan ini membantu Anda melakukan hal berikut ini:

- Terbitkan peristiwa dari jenis `s3:ObjectCreated:*` ke antrean Amazon SQS.

- Terbitkan peristiwa dari jenis `s3:ReducedRedundancyLostObject` ke topik Amazon SNS.

Untuk informasi tentang konfigurasi pemberitahuan, lihat Menggunakan Amazon SQS, Amazon SNS, dan Lambda.

Anda dapat melakukan semua langkah ini menggunakan konsol, tanpa menulis kode apa pun. Selain itu, contoh kode yang menggunakan AWS SDK untuk Java dan .NET juga disediakan untuk membantu Anda menambahkan konfigurasi notifikasi secara terprogram.

Prosedur ini mencakup langkah-langkah berikut:

1. Membuat antrean Amazon SQS.

   Menggunakan konsol Amazon SQS, buat antrean SQS. Anda dapat mengakses pesan apa pun yang dikirimkan Amazon S3 ke antrean secara terprogram. Namun, untuk panduan ini, Anda memverifikasi pesan pemberitahuan di konsol.

   Anda melampirkan kebijakan akses ke antrean untuk memberikan izin kepada Amazon S3 untuk memposting pesan.
2. Buat topik Amazon SNS.

   Dengan konsol Amazon SNS, buat topik SNS dan berlangganan ke topik tersebut. Dengan begitu, setiap peristiwa yang diposting ke sana dikirimkan kepada Anda. Anda menentukan email sebagai protokol komunikasi. Setelah Anda membuat topik, Amazon SNS mengirim email. Anda menggunakan tautan dalam email untuk mengonfirmasi langganan topik.

   Anda melampirkan kebijakan akses ke topik untuk memberikan izin kepada Amazon S3 untuk memposting pesan.
3. Tambahkan konfigurasi pemberitahuan ke bucket.

Langkah 1: Buat antrean Amazon SQS

Ikuti langkah-langkah untuk membuat dan berlangganan antrean Amazon Simple Queue Service (Amazon SQS).

1. Menggunakan konsol Amazon SQS, buat antrean. Untuk instruksi, lihat Memulai dengan Amazon SQS dalam Panduan Pengembang Amazon Simple Queue Service.
2. Ganti kebijakan akses yang terlampir pada antrean dengan kebijakan berikut.

a. Di konsol Amazon SQS, dalam daftar Antrian, pilih nama antrean.

b. Pada tab Kebijakan akses, pilih Edit.

c. Ganti kebijakan akses yang terlampir ke antrean. Di dalamnya, berikan ARN Amazon SQS, nama bucket sumber, dan ID akun pemilik bucket.

```
{
    "Version": "2012-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "example-statement-ID",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "SQS:SendMessage"
            ],
            "Resource": "SQS-queue-ARN",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:*:*:awsexamplebucket1"
                },
                "StringEquals": {
                    "aws:SourceAccount": "bucket-owner-account-id"
                }
            }
        }
    ]
}
```

d. Pilih Simpan.

3. (Opsional) Jika antrean Amazon SQS atau topik Amazon SNS diaktifkan dengan enkripsi sisi server dengan AWS Key Management Service (AWS KMS), tambahkan kebijakan berikut ke kunci terkelola pelanggan enkripsi simetris terkait.

   Anda harus menambahkan kebijakan ke CMK karena Anda tidak dapat memodifikasi kunci AWS terkelola untuk Amazon SQS atau Amazon SNS.

```
{
```

```
    "Version": "2012-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "example-statement-ID",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
    ]
}
```

Untuk informasi selengkapnya tentang penggunaan SSE untuk Amazon SQS dan Amazon SNS AWS KMS with, lihat berikut ini:

- Pengelolaan kunci di Panduan Pengembang Layanan Notifikasi Sederhana Amazon.
- Pengelolaan kunci di Panduan Pengembang Layanan Antrean Sederhana Amazon.

4. Perhatikan antrean ARN.

   Antrean SQS yang Anda buat adalah sumber daya lain di Akun AWS Anda. Amazon Resource Name (ARN) yang unik. Anda membutuhkan ARN ini di langkah berikutnya. ARN memiliki format berikut:

```
arn:aws:sqs:aws-region:account-id:queue-name
```

Langkah 2: Buat Topik Amazon SNS

Ikuti langkah-langkah untuk membuat dan berlangganan topik Amazon SNS.

1. Menggunakan konsol Amazon SNS membuat topik. Untuk instruksi, lihat Membuat topik Amazon SNS dalam Panduan Pengembang Amazon Simple Notification Service.

2. Berlangganan topik tersebut. Untuk latihan ini, gunakan email sebagai protokol komunikasi. Untuk instruksi, lihat Berlangganan topik Amazon SNS dalam Panduan Pengembang Amazon Simple Notification Service.

   Anda mendapatkan email yang meminta Anda untuk mengonfirmasi langganan Anda ke topik tersebut. Konfirmasi langganan.

3. Ganti kebijakan akses yang terlampir pada topik dengan kebijakan berikut. Di dalamnya, berikan ARN topik SNS, nama bucket, dan ID akun pemilik bucket.

```
{
    "Version": "2012-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "Example SNS topic policy",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "SNS:Publish"
            ],
            "Resource": "SNS-topic-ARN",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:*:*:bucket-name"
                },
                "StringEquals": {
                    "aws:SourceAccount": "bucket-owner-account-id"
                }
            }
        }
    ]
}
```

4. Perhatikan topik ARN.

   Topik SNS yang Anda buat adalah sumber daya lain di Anda Akun AWS, dan memiliki ARN yang unik. Anda akan membutuhkan ARN ini di langkah berikutnya. ARN akan memiliki format berikut:

```
arn:aws:sns:aws-region:account-id:topic-name
```

Langkah 3: Menambahkan konfigurasi notifikasi ke bucket Anda

Anda dapat mengaktifkan notifikasi bucket baik dengan menggunakan konsol Amazon S3 atau secara terprogram menggunakan SDK. AWS Memilih salah satu opsi untuk mengonfigurasi notifikasi pada bucket Anda. Bagian ini menyediakan contoh kode menggunakan SDK untuk AWS Java dan .NET.

Opsi A: Mengaktifkan notifikasi pada bucket menggunakan konsol tersebut

Menggunakan konsol Amazon S3, menambahkan konfigurasi notifikasi yang meminta Amazon S3 untuk melakukan hal berikut ini:

- Terbitkan peristiwa Semua objek membuat peristiwa ke antrean Amazon SQS Anda.
- Terbitkan peristiwa Objek dalam RRS hilang ketik topik Amazon SNS Anda.

Setelah Anda menyimpan konfigurasi pemberitahuan, Amazon S3 memposting pesan uji, yang Anda dapatkan melalui email.

Untuk petunjuk, lihat Mengaktifkan dan mengonfigurasi notifikasi peristiwa menggunakan konsol Amazon S3.

Opsi B: Aktifkan notifikasi pada bucket menggunakan AWS SDK

.NET

Contoh kode C# berikut memberikan daftar kode lengkap yang menambahkan konfigurasi pemberitahuan ke bucket. Anda harus memperbarui kode dan memberikan nama bucket dan ARN topik SNS. Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class EnableNotificationsTest
    {
```

```
        private const string bucketName = "*** bucket name ***";
        private const string snsTopic = "*** SNS topic ARN ***";
        private const string sqsQueue = "*** SQS topic ARN ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            EnableNotificationAsync().Wait();
        }

        static async Task EnableNotificationAsync()
        {
            try
            {
               PutBucketNotificationRequest request = new
PutBucketNotificationRequest
                {
                    BucketName = bucketName
                };

                TopicConfiguration c = new TopicConfiguration
                {
                    Events = new List<EventType> { EventType.ObjectCreatedCopy },
                    Topic = snsTopic
                };
                request.TopicConfigurations = new List<TopicConfiguration>();
                request.TopicConfigurations.Add(c);
                request.QueueConfigurations = new List<QueueConfiguration>();
                request.QueueConfigurations.Add(new QueueConfiguration()
                {
                    Events = new List<EventType> { EventType.ObjectCreatedPut },
                    Queue = sqsQueue
                });

                PutBucketNotificationResponse response = await
client.PutBucketNotificationAsync(request);
            }
            catch (AmazonS3Exception e)
            {
```

```
                Console.WriteLine("Error encountered on server. Message:'{0}' ",
 e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown error encountered on server.
 Message:'{0}' ", e.Message);
            }
        }
    }
}
```

Java

Contoh berikut menunjukkan cara menambahkan konfigurasi pemberitahuan ke bucket. Untuk petunjuk tentang cara membuat dan menguji sampel kerja, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.io.IOException;
import java.util.EnumSet;

public class EnableNotificationOnABucket {

    public static void main(String[] args) throws IOException {
        String bucketName = "*** Bucket name ***";
        Regions clientRegion = Regions.DEFAULT_REGION;
        String snsTopicARN = "*** SNS Topic ARN ***";
        String sqsQueueARN = "*** SQS Queue ARN ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();
```

```
            BucketNotificationConfiguration notificationConfiguration = new
 BucketNotificationConfiguration();

            // Add an SNS topic notification.
            notificationConfiguration.addConfiguration("snsTopicConfig",
                    new TopicConfiguration(snsTopicARN,
 EnumSet.of(S3Event.ObjectCreated)));

            // Add an SQS queue notification.
            notificationConfiguration.addConfiguration("sqsQueueConfig",
                    new QueueConfiguration(sqsQueueARN,
 EnumSet.of(S3Event.ObjectCreated)));

            // Create the notification configuration request and set the bucket
 notification
            // configuration.
            SetBucketNotificationConfigurationRequest request = new
 SetBucketNotificationConfigurationRequest(
                    bucketName, notificationConfiguration);
            s3Client.setBucketNotificationConfiguration(request);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Langkah 4: Menguji pengaturan

Sekarang Anda dapat menguji pengaturan dengan mengunggah objek ke bucket Anda dan memverifikasi pemberitahuan peristiwa di konsol Amazon SQS. Untuk instruksi, lihat Menerima Pesan dalam Bagian "Memulai" Panduan Pengembang Amazon Simple Queue Service.

Mengonfigurasi notifikasi peristiwa dengan filter nama kunci objek

Ketika mengkonfigurasi notifikasi peristiwa Amazon S3, Anda harus menentukan mana jenis peristiwa Amazon S3 yang didukung yang menyebabkan Amazon S3 untuk mengirim notifikasi. Jika jenis peristiwa yang Anda tidak tentukan terjadi di bucket S3 Anda, Amazon S3 tidak mengirim notifikasi.

Anda dapat mengonfigurasi pemberitahuan untuk difilter dengan prefiks dan sufiks nama kunci objek. Misalnya, Anda dapat menyiapkan konfigurasi di mana Anda akan dikirimi pemberitahuan hanya ketika file gambar dengan ekstensi nama file "`.jpg`" file ditambahkan ke bucket. Atau, Anda dapat memiliki konfigurasi yang mengirimkan pemberitahuan ke topik Amazon SNS ketika objek dengan awalan `images/` "" ditambahkan ke ember, sambil memiliki pemberitahuan untuk objek dengan awalan `logs/` "" di ember yang sama dikirim ke suatu fungsi. AWS Lambda

> **Note**
>
> Karakter wildcard (“*”) tidak dapat digunakan dalam filter sebagai prefiks atau sufiks. Jika prefiks atau sufiks Anda mengandung spasi, Anda harus menggantinya dengan karakter "+". Jika Anda menggunakan karakter kustom apa pun dalam nilai prefiks atau sufiks, Anda harus memasukkan mereka ke dalam Format yang dikodekan URL (persen-dikodekan). Untuk daftar lengkap karakter kustom yang harus dikonversi ke format yang dikodekan URL saat digunakan dalam prefiks atau sufiks untuk pemberitahuan peristiwa, lihat. Karakter aman

Anda dapat mengatur konfigurasi notifikasi yang menggunakan filter nama kunci objek di konsol Amazon S3. Anda dapat melakukannya dengan menggunakan Amazon S3 API melalui AWS SDK atau REST API secara langsung. Untuk informasi tentang penggunaan UI konsol tersebut untuk mengatur konfigurasi notifikasi pada bucket, lihat Mengaktifkan dan mengonfigurasi notifikasi peristiwa menggunakan konsol Amazon S3.

Amazon S3 menyimpan konfigurasi pemberitahuan sebagai XML di sub-sumber daya notifikasi yang terkait dengan bucket, seperti yang telah dijelaskan di Menggunakan Amazon SQS, Amazon SNS, dan Lambda. Anda menggunakan struktur XML `Filter` untuk menentukan aturan notifikasi yang akan difilter dengan prefiks atau sufiks nama kunci objek. Untuk informasi tentang struktur XML `Filter`, lihat Notifikasi PUT Bucket di Referensi API Amazon Simple Storage Service.

Konfigurasi pemberitahuan yang menggunakan `Filter` tidak dapat menentukan aturan pemfilteran dengan prefiks yang tumpang tindih, sufiks yang tumpang tindih, atau prefiks dan sufiks yang tumpang tindih. Bagian berikut memiliki contoh konfigurasi pemberitahuan yang valid dengan

pemfilteran nama kunci objek. Juga berisi contoh konfigurasi notifikasi yang tidak valid karena prefiks dan sufiks tumpang tindih.

Topik

- Contoh konfigurasi pemberitahuan yang valid dengan pemfilteran nama kunci objek
- Contoh konfigurasi notifikasi dengan prefiks dan sufiks yang tidak valid saling tumpang tindih

Contoh konfigurasi pemberitahuan yang valid dengan pemfilteran nama kunci objek

Konfigurasi notifikasi berikut berisi konfigurasi antrean yang mengidentifikasi antrean Amazon SQS untuk Amazon S3 untuk menerbitkan peristiwa ke jenis `s3:ObjectCreated:Put`. Peristiwa dipublikasikan kapan pun objek yang memiliki prefiks `images/` dan sufiks `jpg` di-PUT ke dalam bucket.

```
<NotificationConfiguration>
  <QueueConfiguration>
      <Id>1</Id>
      <Filter>
          <S3Key>
              <FilterRule>
                  <Name>prefix</Name>
                  <Value>images/</Value>
              </FilterRule>
              <FilterRule>
                  <Name>suffix</Name>
                  <Value>jpg</Value>
              </FilterRule>
          </S3Key>
     </Filter>
     <Queue>arn:aws:sqs:us-west-2:444455556666:s3notificationqueue</Queue>
     <Event>s3:ObjectCreated:Put</Event>
  </QueueConfiguration>
</NotificationConfiguration>
```

Konfigurasi pemberitahuan berikut memiliki beberapa prefiks yang tidak tumpang tindih. Konfigurasi tersebut menentukan bahwa notifikasi untuk permintaan PUT di folder `images/` masuk ke antrean A, sedangkan notifikasi untuk permintaan PUT di `logs/` masuk ke antrean B.

```
<NotificationConfiguration>
  <QueueConfiguration>
```

```
      <Id>1</Id>
      <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>prefix</Name>
                     <Value>images/</Value>
                 </FilterRule>
             </S3Key>
      </Filter>
      <Queue>arn:aws:sqs:us-west-2:444455556666:sqs-queue-A</Queue>
      <Event>s3:ObjectCreated:Put</Event>
   </QueueConfiguration>
   <QueueConfiguration>
      <Id>2</Id>
      <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>prefix</Name>
                     <Value>logs/</Value>
                 </FilterRule>
             </S3Key>
      </Filter>
      <Queue>arn:aws:sqs:us-west-2:444455556666:sqs-queue-B</Queue>
      <Event>s3:ObjectCreated:Put</Event>
   </QueueConfiguration>
</NotificationConfiguration>
```

Konfigurasi notifikasi berikut memiliki beberapa sufiks yang tidak tumpang tindih. Konfigurasi ini menentukan bahwa semua gambar `.jpg` yang baru ditambahkan ke bucket diproses oleh Lambda cloud-function-A, dan semua gambar `.png` yang baru ditambahkan diproses oleh cloud-function-B. Sufiks `.png` dan `.jpg` tidak tumpang tindih, meskipun mereka memiliki huruf terakhir yang sama. Jika string tertentu dapat diakhiri dengan kedua sufiks, kedua sufiks dianggap sebagai tumpang tindih. String tidak dapat diakhiri dengan kedua `.png` dan `.jpg`, sehingga sufiks dalam konfigurasi contoh tidak tumpang tindih dengan sufiks.

```
<NotificationConfiguration>
   <CloudFunctionConfiguration>
      <Id>1</Id>
      <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>suffix</Name>
```

```
                    <Value>.jpg</Value>
                </FilterRule>
            </S3Key>
     </Filter>
     <CloudFunction>arn:aws:lambda:us-west-2:444455556666:cloud-function-A</
CloudFunction>
     <Event>s3:ObjectCreated:Put</Event>
  </CloudFunctionConfiguration>
  <CloudFunctionConfiguration>
     <Id>2</Id>
     <Filter>
            <S3Key>
                <FilterRule>
                    <Name>suffix</Name>
                    <Value>.png</Value>
                </FilterRule>
            </S3Key>
     </Filter>
     <CloudFunction>arn:aws:lambda:us-west-2:444455556666:cloud-function-B</
CloudFunction>
     <Event>s3:ObjectCreated:Put</Event>
  </CloudFunctionConfiguration>
</NotificationConfiguration>
```

Konfigurasi notifikasi Anda yang menggunakan `Filter` tidak dapat menentukan aturan pemfilteran dengan prefiks yang tumpang tindih untuk jenis peristiwa yang sama. Mereka hanya bisa melakukannya, jika prefiks tumpang tindih yang digunakan dengan sufiks yang tidak tumpang tindih. Konfigurasi contoh berikut ini menunjukkan bagaimana objek yang dibuat dengan prefiks yang sama, tetapi sufiks yang tidak tumpang tindih dapat dikirim ke tujuan yang berbeda.

```
<NotificationConfiguration>
  <CloudFunctionConfiguration>
     <Id>1</Id>
     <Filter>
            <S3Key>
                <FilterRule>
                    <Name>prefix</Name>
                    <Value>images</Value>
                </FilterRule>
                <FilterRule>
                    <Name>suffix</Name>
                    <Value>.jpg</Value>
                </FilterRule>
```

```
            </S3Key>
     </Filter>
     <CloudFunction>arn:aws:lambda:us-west-2:444455556666:cloud-function-A</
CloudFunction>
     <Event>s3:ObjectCreated:Put</Event>
  </CloudFunctionConfiguration>
  <CloudFunctionConfiguration>
     <Id>2</Id>
     <Filter>
            <S3Key>
                <FilterRule>
                    <Name>prefix</Name>
                    <Value>images</Value>
                </FilterRule>
                <FilterRule>
                    <Name>suffix</Name>
                    <Value>.png</Value>
                </FilterRule>
            </S3Key>
     </Filter>
     <CloudFunction>arn:aws:lambda:us-west-2:444455556666:cloud-function-B</
CloudFunction>
     <Event>s3:ObjectCreated:Put</Event>
  </CloudFunctionConfiguration>
</NotificationConfiguration>
```

Contoh konfigurasi notifikasi dengan prefiks dan sufiks yang tidak valid saling tumpang tindih

Pada umumnya, konfigurasi pemberitahuan Anda yang menggunakan `Filter` tidak dapat menentukan aturan pemfilteran dengan prefiks yang tumpang tindih, sufiks yang tumpang tindih, atau kombinasi prefiks dan sufiks yang tumpang tindih untuk jenis peristiwa yang sama. Anda dapat memiliki prefiks yang tumpang tindih selama sufiks tidak tumpang tindih. Sebagai contoh, lihat Mengonfigurasi notifikasi peristiwa dengan filter nama kunci objek.

Anda dapat menggunakan filter nama kunci objek yang tumpang tindih dengan jenis peristiwa yang berbeda. Misalnya, Anda dapat membuat konfigurasi notifikasi yang menggunakan prefiks `image/` untuk jenis peristiwa `ObjectCreated:Put`, dan prefiks `image/` untuk jenis peristiwa `ObjectRemoved:*`.

Anda mendapatkan kesalahan jika mencoba menyimpan konfigurasi pemberitahuan yang memiliki filter nama yang tumpang tindih yang tidak valid untuk jenis peristiwa yang sama saat menggunakan

konsol atau API Amazon S3. Bagian ini menunjukkan contoh konfigurasi notifikasi yang tidak valid karena filter nama yang tumpang tindih.

Aturan konfigurasi pemberitahuan apa pun yang ada diasumsikan memiliki prefiks dan sufiks default yang cocok dengan setiap prefiks dan sufiks lain. Konfigurasi pemberitahuan berikut tidak valid karena memiliki prefiks yang tumpang tindih. Secara kustom, prefiks root tumpang tindih dengan prefiks lainnya. Hal yang sama berlaku jika Anda menggunakan sufiks alih-alih prefiks dalam contoh ini. Akhiran root tumpang tindih dengan sufiks lainnya.

```
<NotificationConfiguration>
     <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-notification-one</Topic>
         <Event>s3:ObjectCreated:*</Event>
    </TopicConfiguration>
    <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-notification-two</Topic>
         <Event>s3:ObjectCreated:*</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>prefix</Name>
                     <Value>images</Value>
                 </FilterRule>
            </S3Key>
        </Filter>
    </TopicConfiguration>
</NotificationConfiguration>
```

Konfigurasi pemberitahuan berikut tidak valid karena memiliki sufiks yang tumpang tindih. Jika string tertentu dapat diakhiri dengan kedua sufiks, kedua sufiks dianggap sebagai tumpang tindih. Sebuah string dapat diakhiri dengan `jpg` dan `pg`. Jadi, sufiksnya tumpang tindih. Hal yang sama berlaku untuk prefiks. Jika string tertentu dapat dimulai dengan kedua prefiks, kedua prefiks dianggap tumpang tindih.

```
 <NotificationConfiguration>
     <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-topic-one</Topic>
         <Event>s3:ObjectCreated:*</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
```

```
                    <Name>suffix</Name>
                    <Value>jpg</Value>
                </FilterRule>
            </S3Key>
        </Filter>
    </TopicConfiguration>
    <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-topic-two</Topic>
         <Event>s3:ObjectCreated:Put</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>suffix</Name>
                     <Value>pg</Value>
                 </FilterRule>
             </S3Key>
         </Filter>
    </TopicConfiguration>
</NotificationConfiguration
```

Konfigurasi notifikasi berikut tidak valid karena memiliki prefiks dan sufiks yang tumpang tindih.

```
<NotificationConfiguration>
     <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-topic-one</Topic>
         <Event>s3:ObjectCreated:*</Event>
         <Filter>
             <S3Key>
                 <FilterRule>
                     <Name>prefix</Name>
                     <Value>images</Value>
                 </FilterRule>
                 <FilterRule>
                     <Name>suffix</Name>
                     <Value>jpg</Value>
                 </FilterRule>
             </S3Key>
         </Filter>
    </TopicConfiguration>
    <TopicConfiguration>
         <Topic>arn:aws:sns:us-west-2:444455556666:sns-topic-two</Topic>
         <Event>s3:ObjectCreated:Put</Event>
         <Filter>
```

```
            <S3Key>
                <FilterRule>
                    <Name>suffix</Name>
                    <Value>jpg</Value>
                </FilterRule>
            </S3Key>
        </Filter>
    </TopicConfiguration>
</NotificationConfiguration>
```

Struktur pesan peristiwa

Pesan pemberitahuan yang dikirimkan Amazon S3 untuk menerbitkan kegiatan ada dalam format JSON.

Untuk ikhtisar umum dan petunjuk tentang mengonfigurasi pemberitahuan peristiwa, lihat Notifikasi Peristiwa Amazon S3.

Contoh ini menunjukkan versi 2.2 dari struktur JSON notifikasi peristiwa. Amazon S3 menggunakan versi 2.1, 2.2, dan 2.3 dari struktur peristiwa ini. Amazon S3 menggunakan versi 2.2 untuk pemberitahuan peristiwa replikasi lintas wilayah. Ini menggunakan versi 2.3 untuk Siklus Hidup S3, S3 Intelligent-Tiering, objek ACL, penandaan objek, dan peristiwa penghapusan restorasi objek. Versi ini berisi informasi tambahan kustom untuk operasi ini. Versi 2.2 dan 2.3 sebaliknya kompatibel dengan versi 2.1, yang saat ini digunakan Amazon S3 untuk semua jenis pemberitahuan peristiwa lainnya.

```
{
   "Records":[
      {
         "eventVersion":"2.2",
         "eventSource":"aws:s3",
         "awsRegion":"us-west-2",
         "eventTime":"The time, in ISO-8601 format, for example,
 1970-01-01T00:00:00.000Z, when Amazon S3 finished processing the request",
         "eventName":"event-type",
         "userIdentity":{
            "principalId":"Amazon-customer-ID-of-the-user-who-caused-the-event"
         },
         "requestParameters":{
            "sourceIPAddress":"ip-address-where-request-came-from"
         },
         "responseElements":{
```

```
            "x-amz-request-id":"Amazon S3 generated request ID",
            "x-amz-id-2":"Amazon S3 host that processed the request"
         },
         "s3":{
            "s3SchemaVersion":"1.0",
            "configurationId":"ID found in the bucket notification configuration",
            "bucket":{
               "name":"bucket-name",
               "ownerIdentity":{
                  "principalId":"Amazon-customer-ID-of-the-bucket-owner"
               },
               "arn":"bucket-ARN"
            },
            "object":{
               "key":"object-key",
               "size":"object-size in bytes",
               "eTag":"object eTag",
               "versionId":"object version if bucket is versioning-enabled, otherwise
 null",
               "sequencer": "a string representation of a hexadecimal value used to
 determine event sequence, only used with PUTs and DELETEs"
            }
         },
         "glacierEventData": {
            "restoreEventData": {
               "lifecycleRestorationExpiryTime": "The time, in ISO-8601 format, for
 example, 1970-01-01T00:00:00.000Z, of Restore Expiry",
               "lifecycleRestoreStorageClass": "Source storage class for restore"
            }
         }
      }
   ]
}
```

Perhatikan hal berikut tentang struktur pesan peristiwa:

- Nilai kunci `eventVersion` mengandung versi besar dan kecil, dalam bentuk `<major>.<minor>`.

  Versi utama bertambah jika Amazon S3 membuat perubahan pada struktur peristiwa yang tidak kompatibel dengan versi sebelumnya. Ini termasuk menghapus bidang JSON yang sudah ada atau mengubah bagaimana isi bidang direpresentasikan (misalnya, format tanggal).

Versi minor ditingkatkan jika Amazon S3 menambahkan bidang baru ke struktur peristiwa. Hal ini dapat terjadi jika informasi baru diberikan untuk beberapa atau semua peristiwa yang ada. Ini mungkin juga terjadi jika informasi baru hanya diberikan pada jenis peristiwa yang baru diperkenalkan. Aplikasi sebaiknya mengabaikan bidang baru untuk tetap kompatibel dengan versi kecil struktur peristiwa baru.

Jika jenis peristiwa baru diperkenalkan tetapi struktur peristiwa tidak dimodifikasi, versi peristiwa tidak akan berubah.

Untuk memastikan aplikasi Anda dapat menguraikan struktur peristiwa dengan benar, kami menyarankan Anda melakukan perbandingan yang setara dengan nomor versi besar. Untuk memastikan bahwa bidang yang diharapkan oleh aplikasi Anda ada, kami juga merekomendasikan melakukan perbandingan greater-than-or-equal -to pada versi minor.

- Angka `eventName` mereferensikan daftar jenis notifikasiperistiwa tetapi tidak berisi prefiks `s3:`.
- Nilai `responseElements` kunci berguna jika Anda ingin melacak permintaan dengan menindaklanjuti AWS Support. Baik `x-amz-request-id` maupun `x-amz-id-2` membantu Amazon S3 menelusuri permintaan individu. Nilai-nilai ini sama dengan nilai yang dikembalikan Amazon S3 sebagai respons terhadap permintaan yang memulai peristiwa. Ini agar mereka dapat digunakan untuk mencocokkan peristiwa dengan permintaan.
- Kunci `s3` memberikan informasi tentang bucket dan objek yang terlibat dalam peristiwa tersebut. Nilai nama kunci objek dienkode URL. Misalnya, "red flower.jpg" menjadi "red+flower.jpg" (Amazon S3 mengembalikan "`application/x-www-form-urlencoded`" sebagai jenis konten di dalam respons tersebut).
- Kunci `sequencer` memberikan cara untuk menentukan urutan peristiwa. Pemberitahuan peristiwa tidak dijamin akan tiba dalam urutan yang sama dengan peristiwa yang terjadi. Namun, pemberitahuan dari peristiwa yang membuat objek (PUT) dan menghapus objek berisi file `sequencer`. Ini dapat digunakan untuk menentukan urutan peristiwa untuk kunci objek tertentu.

  Jika Anda membandingkan string `sequencer` dari dua notifikasi peristiwa pada kunci objek yang sama, pemberitahuan peristiwa dengan nilai heksadesimal `sequencer` yang lebih besar adalah peristiwa yang terjadi kemudian. Jika Anda menggunakan notifikasi peristiwa untuk memelihara basis data atau indeks terpisah dari objek Amazon S3, kami sarankan Anda membandingkan dan menyimpan nilai `sequencer` saat Anda memproses setiap pemberitahuan peristiwa.

  Perhatikan hal berikut:

  - Anda tidak dapat menggunakan `sequencer` untuk menentukan urutan peristiwa dengan kunci objek yang berbeda.
  - Urutan dapat memiliki panjang berbeda. Jadi, untuk membandingkan nilai-nilai ini, pertama-tama masukkan nilai yang lebih pendek dengan nol, dan kemudian melakukan perbandingan leksikografis.
- Kunci `glacierEventData` hanya terlihat untuk peristiwa `s3:ObjectRestore:Completed`.
- Kunci `restoreEventData` berisi atribut yang terkait dengan permintaan pemulihan Anda.
- Kunci `replicationEventData` hanya terlihat untuk peristiwa replikasi.
- Kunci `intelligentTieringEventData` hanya terlihat untuk peristiwa S3 Intelligent-Tiering.
- Kunci `lifecycleEventData` hanya terlihat untuk peristiwa transisi Siklus Hidup S3.

Contoh pesan

Berikut ini adalah contoh dari pesan notifikasi peristiwa Amazon S3.

Olahpesan pengujian Amazon S3

Setelah mengonfigurasi notifikasi peristiwa di bucket, Amazon S3 mengirimkan pesan pengujian berikut.

```
{
   "Service":"Amazon S3",
   "Event":"s3:TestEvent",
   "Time":"2014-10-13T15:57:02.089Z",
   "Bucket":"bucketname",
   "RequestId":"5582815E1AEA5ADF",
   "HostId":"8cLeGAmw098X5cv4Zkwcmo8vvZa3eH3eKxsPzbB9wrR+YstdA6Knx4Ip8EXAMPLE"
}
```

Contoh pesan ketika sebuah objek dibuat menggunakan permintaan PUT

Olahpesan berikut adalah contoh pesan yang dikirimkan Amazon S3 untuk menerbitkan peristiwa `s3:ObjectCreated:Put`.

```
{
   "Records":[
      {
         "eventVersion":"2.1",
         "eventSource":"aws:s3",
```

```
          "awsRegion":"us-west-2",
          "eventTime":"1970-01-01T00:00:00.000Z",
          "eventName":"ObjectCreated:Put",
          "userIdentity":{
             "principalId":"AIDAJDPLRKLG7UEXAMPLE"
          },
          "requestParameters":{
             "sourceIPAddress":"127.0.0.1"
          },
          "responseElements":{
             "x-amz-request-id":"C3D13FE58DE4C810",
             "x-amz-id-2":"FMyUVURIY8/IgAtTv8xRjskZQpcIZ9KG4V5Wp6S7S/
JRWeUWerMUE5JgHvANOjpD"
          },
          "s3":{
             "s3SchemaVersion":"1.0",
             "configurationId":"testConfigRule",
             "bucket":{
                "name":"mybucket",
                "ownerIdentity":{
                   "principalId":"A3NL1KOZZKExample"
                },
                "arn":"arn:aws:s3:::mybucket"
             },
             "object":{
                "key":"HappyFace.jpg",
                "size":1024,
                "eTag":"d41d8cd98f00b204e9800998ecf8427e",
                "versionId":"096fKKXTRTtl3on89fVO.nfljtsv6qko",
                "sequencer":"0055AED6DCD90281E5"
             }
          }
       }
    ]
}
```

Untuk definisi setiap prefiks identifikasi IAM (misalnya, AIDA, AROA, AGPA), lihat Pengidentifikasi IAM di Panduan Pengguna IAM.

Menggunakan EventBridge

Amazon S3 dapat mengirim acara ke Amazon EventBridge setiap kali peristiwa tertentu terjadi di bucket Anda. Tidak seperti tujuan lain, Anda tidak perlu memilih jenis peristiwa mana yang ingin

Anda kirimkan. Setelah EventBridge diaktifkan, semua acara di bawah ini dikirim ke EventBridge. Anda dapat menggunakan EventBridge aturan untuk merutekan acara ke target tambahan. Berikut ini mencantumkan peristiwa yang dikirimkan Amazon S3. EventBridge

Jenis peristiwa	Deskripsi
Objek Dibuat	Sebuah objek telah dibuat. Bidang alasan dalam struktur pesan peristiwa menunjukkan S3 API mana yang digunakan untuk membuat objek: PutObject, POST Object, CopyObject, atau CompleteMultipartUpload.
Objek Dihapus (DeleteObject) Objek Dihapus (Kedaluwarsa siklus hidup)	Sebuah objek telah dihapus. Saat objek dihapus menggunakan panggilan API S3, bidang alasan disetel ke DeleteObject. Saat objek dihapus oleh aturan kedaluwarsa Siklus Hidup S3, bidang alasan disetel ke Kedaluwarsa Siklus Hidup. Untuk informasi selengkapnya, lihat Mengakhiri objek. Ketika objek yang tidak berversi dihapus, atau objek berversi dihapus secara permanen, bidang jenis penghapusan diatur ke Dihapus Secara Permanen. Saat penanda penghapusan dibuat untuk objek versi, bidang jenis penghapusan diatur ke Delete Marker Created. Untuk informasi selengkapnya, lihat Menghapus versi objek dari bucket dengan dukungan Penentuan Versi.
Pemulihan Objek Dimulai	Pemulihan objek dimulai dari kelas penyimpanan S3 Glacier atau S3 Glacier Deep Archive storage class atau from S3 Intelligent-Tiering Archive Access atau Deep Archive Access tier. Untuk informasi selengkapnya, lihat Bekerja dengan objek yang diarsipkan.
Pemulihan Objek Selesai	Pemulihan objek selesai.

Jenis peristiwa	Deskripsi
Pemulihan Objek Kedaluwarsa	Salinan sementara dari objek yang dipulihkan dari S3 Glacier atau S3 Glacier Deep Archive telah kedaluwarsa dan dihapus.
Kelas Penyimpanan Objek Berubah	Sebuah objek dialihkan ke kelas penyimpanan yang berbeda. Untuk informasi selengkapnya, lihat Transisi objek menggunakan Siklus Hidup Amazon S3.
Tingkat Akses Objek Berubah	Sebuah objek dialihkan ke tingkat S3 Intelligent-Tiering Archive Access atau tingkat Akses Arsip Dalam. Untuk informasi selengkapnya, lihat Amazon S3 Intelligent-Tiering.
Objek ACL Diperbarui	Daftar kontrol akses objek (ACL) ditetapkan menggunakan PutObject ACL. Peristiwa tidak dihasilkan ketika permintaannya tidak menghasilkan perubahan pada ACL objek. Untuk informasi selengkapnya, lihat Gambaran umum daftar kontrol akses (ACL).
Tag Objek Ditambahkan	Satu set tag ditambahkan ke objek menggunakan PutObjectTagging. Untuk informasi selengkapnya, lihat Mengategorikan penyimpanan Anda menggunakan tag.
Tag Objek Dihapus	Semua tag telah dihapus dari objek menggunakan DeleteObjectTagging. Untuk informasi selengkapnya, lihat Mengategorikan penyimpanan Anda menggunakan tag.

> **Note**
>
> Untuk informasi selengkapnya tentang cara memetakan jenis peristiwa Amazon S3 ke jenis EventBridge acara, lihat. EventBridge Pemetaan dan pemecahan masalah Amazon

Anda dapat menggunakan Pemberitahuan Acara Amazon S3 EventBridge untuk menulis aturan yang mengambil tindakan saat peristiwa terjadi di bucket Anda. Misalnya, Anda dapat mengirimi

Anda pemberitahuan. Untuk informasi selengkapnya, lihat Apa yang ada EventBridge di Panduan EventBridge Pengguna Amazon.

Untuk informasi selengkapnya tentang tindakan dan tipe data yang dapat berinteraksi dengan menggunakan EventBridge API, lihat Referensi Amazon EventBridge API di Referensi Amazon EventBridge API.

Untuk informasi tentang harga, lihat EventBridge harga Amazon.

Topik

- EventBridge Izin Amazon
- Mengaktifkan Amazon EventBridge
- EventBridge struktur pesan acara
- EventBridge Pemetaan dan pemecahan masalah Amazon

EventBridge Izin Amazon

Amazon S3 tidak memerlukan izin tambahan apa pun untuk mengirimkan acara ke Amazon. EventBridge

Mengaktifkan Amazon EventBridge

Anda dapat mengaktifkan Amazon EventBridge menggunakan konsol S3, AWS Command Line Interface (AWS CLI), atau Amazon S3 REST API.

Menggunakan konsol S3

Untuk mengaktifkan pengiriman EventBridge acara di konsol S3.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di Bucket pilih nama bucket yang ingin Anda aktifkan untuk peristiwa.
3. Pilih Properti.
4. Arahkan ke bagian Pemberitahuan Acara dan temukan EventBridge subbagian Amazon. Pilih Edit.
5. Di bawah Kirim notifikasi ke Amazon EventBridge untuk semua acara di bucket ini pilih Aktif.

> **Note**
>
> Setelah Anda mengaktifkan EventBridge, dibutuhkan sekitar lima menit agar perubahan diterapkan.

Menggunakan AWS CLI

Contoh berikut membuat konfigurasi notifikasi bucket untuk bucket `DOC-EXAMPLE-BUCKET1` dengan Amazon EventBridge diaktifkan.

```
aws s3api put-bucket-notification-configuration --bucket example-s3-bucket1 --
notification-configuration='{ "EventBridgeConfiguration": {} }'
```

Penggunaan API REST

Anda dapat mengaktifkan Amazon secara terprogram EventBridge di bucket dengan memanggil Amazon S3 REST API. Untuk informasi selengkapnya lihat, lihat PutBucketNotificationConfigurationdi Referensi API Amazon Simple Storage Service.

Contoh berikut menunjukkan XHTML yang digunakan untuk membuat konfigurasi notifikasi bucket dengan Amazon EventBridge diaktifkan.

```
<NotificationConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <EventBridgeConfiguration>
  </EventBridgeConfiguration>
</NotificationConfiguration>
```

Membuat EventBridge aturan

Setelah diaktifkan, Anda dapat membuat EventBridge aturan Amazon untuk tugas tertentu. Misalnya, Anda dapat mengirim notifikasi email ketika objek dibuat. Untuk tutorial selengkapnya, lihat Tutorial: Mengirim pemberitahuan saat objek Amazon S3 dibuat di EventBridge Panduan Pengguna Amazon.

EventBridge struktur pesan acara

Pesan pemberitahuan yang dikirimkan Amazon S3 untuk menerbitkan kegiatan ada dalam format JSON. Saat Amazon S3 mengirim acara ke Amazon EventBridge, bidang berikut hadir.

- version—Saat ini adalah 0 (nol) untuk semua peristiwa.
- id—Versi 4 UID yang dihasilkan untuk setiap peristiwa.
- detail-type—Jenis peristiwa yang sedang dikirim. Lihat Menggunakan EventBridge untuk daftar jenis peristiwa.
- source—Mengidentifikasi layanan yang menghasilkan peristiwa.
- account—ID 12 digit Akun AWS dari pemilik bucket.
- time—Waktu peristiwa terjadi.
- region—Mengidentifikasi Wilayah AWS dari bucket tersebut.
- resources—Array JSON yang berisi Amazon Resource Name (ARN) pada bucket.
- detail—Objek JSON yang berisi informasi tentang peristiwa. Untuk informasi selengkapnya tentang apa saja yang dapat disertakan dalam bidang ini, lihat Kolom detail pesan peristiwa.

Contoh struktur pesan peristiwa

Berikut ini adalah contoh dari beberapa pesan notifikasi acara Amazon S3 yang dapat dikirim ke Amazon. EventBridge

Objek dibuat

```
{
  "version": "0",
  "id": "17793124-05d4-b198-2fde-7ededc63b103",
  "detail-type": "Object Created",
  "source": "aws.s3",
  "account": "111122223333",
  "time": "2021-11-12T00:00:00Z",
  "region": "ca-central-1",
  "resources": [
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
  ],
  "detail": {
    "version": "0",
    "bucket": {
      "name": "DOC-EXAMPLE-BUCKET1"
    },
    "object": {
      "key": "example-key",
      "size": 5,
      "etag": "b1946ac92492d2347c6235b4d2611184",
```

```
      "version-id": "IYV3p45BT0ac8hjHg1houSdS1a.Mro8e",
      "sequencer": "617f08299329d189"
    },
    "request-id": "N4N7GDK58NMKJ12R",
    "requester": "123456789012",
    "source-ip-address": "1.2.3.4",
    "reason": "PutObject"
  }
}
```

Objek dihapus (menggunakan DeleteObject)

```
{
  "version": "0",
  "id": "2ee9cc15-d022-99ea-1fb8-1b1bac4850f9",
  "detail-type": "Object Deleted",
  "source": "aws.s3",
  "account": "111122223333",
  "time": "2021-11-12T00:00:00Z",
  "region": "ca-central-1",
  "resources": [
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
  ],
  "detail": {
    "version": "0",
    "bucket": {
      "name": "DOC-EXAMPLE-BUCKET1"
    },
    "object": {
      "key": "example-key",
      "etag": "d41d8cd98f00b204e9800998ecf8427e",
      "version-id": "1QW9g1Z99LUNbvaaYVpW9xDlOLU.qxgF",
      "sequencer": "617f0837b476e463"
    },
    "request-id": "0BH729840619AG5K",
    "requester": "123456789012",
    "source-ip-address": "1.2.3.4",
    "reason": "DeleteObject",
    "deletion-type": "Delete Marker Created"
  }
}
```

Objek dihapus (menggunakan kedaluwarsa siklus hidup)

```
{
  "version": "0",
  "id": "ad1de317-e409-eba2-9552-30113f8d88e3",
  "detail-type": "Object Deleted",
  "source": "aws.s3",
  "account": "111122223333",
  "time": "2021-11-12T00:00:00Z",
  "region": "ca-central-1",
  "resources": [
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
  ],
  "detail": {
    "version": "0",
    "bucket": {
      "name": "DOC-EXAMPLE-BUCKET1"
    },
    "object": {
      "key": "example-key",
      "etag": "d41d8cd98f00b204e9800998ecf8427e",
      "version-id": "mtB0cV.jejK63XkRNceanNMC.qXPWLeK",
      "sequencer": "617b398000000000"
    },
    "request-id": "20EB74C14654DC47",
    "requester": "s3.amazonaws.com",
    "reason": "Lifecycle Expiration",
    "deletion-type": "Delete Marker Created"
  }
}
```

Pemulihan objek selesai

```
{
  "version": "0",
  "id": "6924de0d-13e2-6bbf-c0c1-b903b753565e",
  "detail-type": "Object Restore Completed",
  "source": "aws.s3",
  "account": "111122223333",
  "time": "2021-11-12T00:00:00Z",
  "region": "ca-central-1",
  "resources": [
```

```
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
  ],
  "detail": {
    "version": "0",
    "bucket": {
      "name": "DOC-EXAMPLE-BUCKET1"
    },
    "object": {
      "key": "example-key",
      "size": 5,
      "etag": "b1946ac92492d2347c6235b4d2611184",
      "version-id": "KKsjUC1.6gIjqtvhfg5AdMI0eCePIiT3"
    },
    "request-id": "189F19CB7FB1B6A4",
    "requester": "s3.amazonaws.com",
    "restore-expiry-time": "2021-11-13T00:00:00Z",
    "source-storage-class": "GLACIER"
  }
}
```

Kolom detail pesan peristiwa

Bidang detail berisi objek JSON dengan informasi tentang peristiwa tersebut. Bidang berikut mungkin ada di bidang detail.

- version—Saat ini adalah 0 (nol) untuk semua peristiwa.
- bucket—Informasi tentang bucket Amazon S3 yang terlibat dalam peristiwa tersebut.
- object—Informasi tentang objek Amazon S3 yang terlibat dalam peristiwa tersebut.
- request-id—ID permintaan dalam respons S3.
- pemohon — Akun AWS ID atau prinsip AWS layanan pemohon.
- source-ip-address— Alamat IP sumber permintaan S3. Hanya terjadi untuk peristiwa yang dipicu oleh permintaan S3.
- alasan - Untuk peristiwa Object Created, API S3 digunakan untuk membuat objek: PutObject, POST Object, CopyObject, atau CompleteMultipartUpload. Untuk peristiwa Object Deleted, ini diatur ke DeleteObjectsaat objek dihapus oleh panggilan API S3, atau Kedaluwarsa Siklus Hidup saat objek dihapus oleh aturan kedaluwarsa Siklus Hidup S3. Untuk informasi selengkapnya, lihat Mengakhiri objek.

- deletion-type—Untuk peristiwa Object Deleted, ketika objek yang tidak berversi dihapus, atau objek berversi dihapus secara permanen, ini diatur ke Permanently Deleted. Saat penanda penghapusan dibuat untuk objek versi, ini diatur ke Hapus Penanda yang Dibuat. Untuk informasi selengkapnya, lihat Menghapus versi objek dari bucket dengan dukungan Penentuan Versi.
- restore-expiry-time— Untuk acara Object Restore Selesai, waktu ketika salinan sementara objek akan dihapus dari S3. Untuk informasi selengkapnya, lihat Bekerja dengan objek yang diarsipkan.
- source-storage-class— Untuk peristiwa Object Restore Inisiated dan Object Restore Completed, kelas penyimpanan objek dipulihkan. Untuk informasi selengkapnya, lihat Bekerja dengan objek yang diarsipkan.
- destination-storage-class— Untuk Object Storage Class Changed event, kelas penyimpanan baru dari objek. Untuk informasi selengkapnya, lihat Transisi objek menggunakan Siklus Hidup Amazon S3.
- destination-access-tier— Untuk acara Object Access Tier Changed, tingkat akses baru objek. Untuk informasi selengkapnya, lihat Amazon S3 Intelligent-Tiering.

EventBridge Pemetaan dan pemecahan masalah Amazon

Tabel berikut menjelaskan bagaimana jenis peristiwa Amazon S3 dipetakan ke jenis peristiwa Amazon EventBridge .

Jenis peristiwa S3	Jenis EventBridge detail Amazon
ObjectCreated:Masukan ObjectCreated:Posting ObjectCreated:Salin ObjectCreated:CompleteMultipartUpload	Objek Dibuat
ObjectRemoved:Hapus ObjectRemoved:DeleteMarkerCreated LifecycleExpiration:Hapus	Object Deleted

Jenis peristiwa S3	Jenis EventBridge detail Amazon
LifecycleExpiration:DeleteMarkerCreated	
ObjectRestore:Posting	Pemulihan Objek Dimulai
ObjectRestore:Selesai	Pemulihan Objek Selesai
ObjectRestore:Hapus	Pemulihan Objek Kedaluwarsa
LifecycleTransition	Kelas Penyimpanan Objek Berubah
IntelligentTiering	Tingkat Akses Objek Berubah
ObjectTagging:Masukan	Tag Objek Ditambahkan
ObjectTagging:Hapus	Tag Objek Dihapus
ObjectAcl:Masukan	Objek ACL Diperbarui

EventBridge Pemecahan masalah Amazon

Untuk informasi tentang cara memecahkan masalah EventBridge, lihat Memecahkan Masalah Amazon EventBridge di Panduan Pengguna Amazon EventBridge .

Menggunakan analitik dan wawasan

Anda dapat menggunakan analitik dan wawasan di Amazon S3 untuk memahami, menganalisis, dan mengoptimalkan penggunaan penyimpanan Anda. Untuk informasi selengkapnya, lihat topik di bawah.

Topik

- Analitik Amazon S3–Analisis Kelas Penyimpanan
- Menilai aktivitas penyimpanan dan penggunaan Anda dengan Amazon S3 Storage Lens
- Melacak permintaan Amazon S3 menggunakan AWS X-Ray

Analitik Amazon S3–Analisis Kelas Penyimpanan

Dengan menggunakan analitik Amazon S3 Analisis Kelas Penyimpanan Anda dapat menganalisis pola akses penyimpanan untuk membantu memutuskan kapan harus memindahkan data yang tepat ke kelas penyimpanan yang tepat. Fitur analitik Amazon S3 yang baru ini mengamati pola akses data untuk membantu Anda menentukan kapan harus memindahkan penyimpanan STANDARD yang jarang diakses ke kelas penyimpanan STANDARD_IA (IA, untuk akses yang jarang). Untuk informasi selengkapnya tentang kelas penyimpanan, lihat Menggunakan kelas penyimpanan Amazon S3.

Setelah analisis kelas penyimpanan mengamati pola akses yang jarang dari kumpulan data yang difilter selama periode waktu tertentu, Anda dapat menggunakan hasil analisis untuk membantu Anda meningkatkan konfigurasi siklus hidup. Anda dapat mengonfigurasi analisis kelas penyimpanan untuk menganalisis semua objek dalam bucket. Atau, Anda dapat mengonfigurasi filter untuk mengelompokkan objek yang akan dianalisis berdasarkan prefiks umum (yaitu, objek yang memiliki nama yang dimulai dengan string umum), berdasarkan tanda objek, atau berdasarkan prefiks dan tanda. Anda akan mengetahui bahwa pemfilteran berdasarkan kelompok objek merupakan cara terbaik untuk memanfaatkan analisis kelas penyimpanan.

> **Important**
>
> Analisis kelas penyimpanan hanya memberikan rekomendasi untuk kelas Standar ke Standar IA.

Anda dapat memiliki banyak filter analisis kelas penyimpanan per bucket, hingga 1.000, dan setiap filter akan memberikan hasil analisis yang terpisah. Konfigurasi beberapa filter memungkinkan Anda menganalisis kelompok objek tertentu untuk meningkatkan konfigurasi siklus hidup yang memindahkan objek ke STANDARD_IA.

Analisis kelas penyimpanan memberikan visualisasi penggunaan penyimpanan di konsol Amazon S3 yang diperbarui setiap hari. Anda juga dapat mengekspor data penggunaan harian ini ke bucket S3 dan melihatnya dalam aplikasi spreadsheet, atau dengan alat intelijen bisnis, seperti Amazon. QuickSight

Anda akan dikenakan biaya yang terkait dengan analisis kelas penyimpanan. Untuk informasi harga, lihat Manajemen dan replikasi Harga Amazon S3.

Topik

- Bagaimana cara menyiapkan analisis kelas penyimpanan?
- Bagaimana cara menggunakan analisis kelas penyimpanan?
- Bagaimana cara mengekspor data analisis kelas penyimpanan?
- Mengonfigurasi analisis kelas penyimpanan

Bagaimana cara menyiapkan analisis kelas penyimpanan?

Anda mengatur analisis kelas penyimpanan dengan mengonfigurasi data objek yang ingin dianalisis. Anda dapat mengonfigurasi analisis kelas penyimpanan untuk melakukan hal berikut:

- Menganalisis seluruh konten bucket.

  Anda akan menerima analisis untuk semua objek dalam bucket.
- Menganalisis objek yang dikelompokkan bersama berdasarkan prefiks dan tanda.

  Anda dapat mengonfigurasi filter yang mengelompokkan objek untuk analisis berdasarkan prefiks, tanda objek, atau kombinasi prefiks dan tanda. Anda akan menerima analisis terpisah untuk setiap filter yang dikonfigurasi. Anda dapat memiliki banyak konfigurasi filter per bucket, hingga 1.000.
- Mengekspor data analisis.

  Saat mengonfigurasi analisis kelas penyimpanan untuk bucket atau filter, Anda dapat memilih untuk mengekspor data analisis ke file setiap hari. Analisis untuk hari itu ditambahkan ke file agar membentuk log analisis historis untuk filter yang dikonfigurasi. File ini diperbarui setiap hari di

tujuan pilihan Anda. Saat memilih data yang akan diekspor, Anda menentukan bucket tujuan dan prefiks tujuan opsional tempat file akan ditulis.

Anda dapat menggunakan konsol Amazon S3, REST API, AWS CLI atau AWS SDK untuk mengonfigurasi analisis kelas penyimpanan.

- Untuk informasi tentang cara mengonfigurasi analisis kelas penyimpanan di konsol Amazon S3, lihat Mengonfigurasi analisis kelas penyimpanan.
- Untuk menggunakan Amazon S3 API, gunakan PutBucketAnalyticsConfigurationREST API, atau yang setara, dari AWS CLI atau AWS SDK.

Bagaimana cara menggunakan analisis kelas penyimpanan?

Anda menggunakan analisis kelas penyimpanan untuk mengamati pola akses data dari waktu ke waktu untuk mengumpulkan informasi guna membantu Anda meningkatkan manajemen siklus hidup penyimpanan STANDARD_IA. Setelah mengonfigurasi filter, Anda akan mulai melihat analisis data berdasarkan filter di konsol Amazon S3 dalam 24 hingga 48 jam. Namun, analisis kelas penyimpanan mengamati pola akses dari set data yang difilter selama 30 hari atau lebih guna mengumpulkan informasi untuk digunakan dalam analisis sebelum memberikan hasil. Analisis terus berjalan setelah hasil awal dan akan memperbarui hasilnya seiring perubahan pola akses

Saat Anda pertama kali mengonfigurasi filter, konsol Amazon S3 mungkin memerlukan waktu untuk menganalisis data Anda.

Analisis kelas penyimpanan mengamati pola akses dari kumpulan data yang difilter selama 30 hari atau lebih guna mengumpulkan informasi yang cukup untuk analisis. Setelah analisis kelas penyimpanan mengumpulkan informasi yang cukup, Anda akan melihat pesan di konsol Amazon S3 bahwa analisis telah selesai.

Saat melakukan analisis untuk objek yang jarang diakses, analisis kelas penyimpanan melihat kumpulan objek yang dikelompokkan bersama berdasarkan umur sejak diunggah ke Amazon S3. Analisis kelas penyimpanan menentukan apakah kelompok umur jarang diakses dengan melihat faktor-faktor berikut untuk kumpulan data yang difilter:

- Objek dalam kelas penyimpanan STANDARD yang lebih besar dari 128 KB.
- Berapa rata-rata total penyimpanan yang Anda miliki per kelompok umur.
- Rata-rata jumlah byte yang ditransfer keluar (bukan frekuensi) per kelompok umur.

- Data ekspor analitik hanya mencakup permintaan dengan data yang relevan dengan analisis kelas penyimpanan. Hal ini dapat menyebabkan perbedaan dalam jumlah permintaan, dan total unggahan serta byte permintaan dibandingkan dengan yang ditunjukkan dalam metrik penyimpanan atau yang dilacak oleh sistem internal Anda sendiri.
- Permintaan GET dan PUT yang gagal tidak dihitung untuk analisis. Namun, Anda dapat melihat permintaan gagal dalam metrik penyimpanan.

Berapa Banyak Penyimpanan yang Saya Ambil?

Konsol Amazon S3 memetakan jumlah penyimpanan dalam kumpulan data yang difilter yang telah diambil untuk periode observasi.

Berapa Persentase Penyimpanan yang Saya Ambil?

Konsol Amazon S3 juga memetakan persentase penyimpanan dalam kumpulan data yang difilter yang telah diambil untuk periode observasi.

Seperti yang telah disebutkan sebelumnya dalam topik ini, saat Anda melakukan analisis untuk objek yang jarang diakses, analisis kelas penyimpanan melihat kumpulan objek yang telah difilter yang dikelompokkan berdasarkan umur sejak diunggah ke Amazon S3. Analisis kelas penyimpanan menggunakan kelompok umur objek yang sudah ditentukan berikut ini:

- Objek Amazon S3 dengan umur kurang dari 15 hari
- Objek Amazon S3 dengan umur 15-29 hari
- Objek Amazon S3 dengan umur 30-44 hari
- Objek Amazon S3 dengan umur 45-59 hari
- Objek Amazon S3 dengan umur 60-74 hari
- Objek Amazon S3 dengan umur 75-89 hari
- Objek Amazon S3 dengan umur 90-119 hari
- Objek Amazon S3 dengan umur 120-149 hari
- Objek Amazon S3 dengan umur 150-179 hari
- Objek Amazon S3 dengan umur 180-364 hari
- Objek Amazon S3 dengan umur 365-729 hari
- Objek Amazon S3 dengan umur 730 hari dan lebih lama

Biasanya dibutuhkan waktu sekitar 30 hari untuk mengamati pola akses untuk mengumpulkan informasi yang cukup sebagai hasil analisis. Mungkin dibutuhkan waktu lebih dari 30 hari, bergantung pada pola akses unik dari data Anda. Namun, setelah mengonfigurasi filter, Anda akan mulai melihat analisis data berdasarkan filter di konsol Amazon S3 dalam 24 hingga 48 jam. Anda dapat melihat analisis akses objek yang dipecah berdasarkan kelompok umur objek di konsol Amazon S3 setiap hari.

Berapa Banyak Penyimpanan Saya yang Jarang Diakses?

Konsol Amazon S3 menunjukkan pola akses yang dikelompokkan berdasarkan kelompok umur objek yang telah ditentukan sebelumnya. Keterangan Sering diakses atau Jarang diakses yang ditampilkan bertujuan untuk membantu Anda dalam proses pembuatan siklus hidup.

Bagaimana cara mengekspor data analisis kelas penyimpanan?

Anda dapat memilih agar analisis kelas penyimpanan mengekspor laporan analisis ke dalam file datar yang nilainya dipisahkan koma (CSV). Laporan diperbarui setiap hari dan didasarkan pada filter kelompok umur objek yang Anda konfigurasi. Saat menggunakan konsol Amazon S3, Anda dapat memilih opsi laporan ekspor ketika membuat filter. Saat memilih ekspor data, Anda harus menentukan bucket tujuan dan prefiks tujuan opsional tempat file ditulis. Anda dapat mengekspor data ke bucket tujuan di akun lain. Bucket tujuan harus berada di wilayah yang sama dengan bucket yang Anda konfigurasikan untuk dianalisis.

Anda harus membuat kebijakan bucket di bucket tujuan untuk memberikan izin ke Amazon S3 guna memverifikasi Akun AWS apa yang memiliki bucket dan menulis objek ke bucket di lokasi yang ditentukan. Untuk contoh kebijakan, lihat Berikan izin untuk Inventaris S3 dan analitik S3.

Setelah mengonfigurasi laporan analisis kelas penyimpanan, Anda akan mulai mendapatkan laporan yang diekspor setiap hari setelah 24 jam. Setelah itu, Amazon S3 akan terus memantau dan menyediakan ekspor harian.

Anda dapat membuka file CSV dalam aplikasi spreadsheet atau mengimpor file ke aplikasi lain seperti Amazon. QuickSight Untuk informasi tentang penggunaan file Amazon S3 dengan Amazon QuickSight, lihat Membuat Kumpulan Data Menggunakan File Amazon S3 di Panduan Pengguna Amazon QuickSight .

Data dalam file yang diekspor diurutkan berdasarkan tanggal dalam kelompok umur objek seperti yang ditunjukkan dalam contoh berikut. Jika kelas penyimpanannya STANDAR, baris juga berisi data untuk kolom `ObjectAgeForSIATransition` dan `RecommendedObjectAgeForSIATransition`.

Date	ConfigId	Filter	StorageClass	ObjectAge	ObjectCount	DataUploaded_MB	Storage_MB	DataRetrieved_MB	GetRequestCount	CumulativeAccessRatio	ObjectAgeForSIATransition	RecommendedObjectAgeForSIATransition
8/17/2021	SalesMaterial	SalesMaterial	STANDARD	000-014			0.4313			0		
9/2/2021	SalesMaterial	SalesMaterial	STANDARD	000-014						0.04096734		
8/22/2021	SalesMaterial	SalesMaterial	STANDARD	000-014			0.4313			0		
8/25/2021	SalesMaterial	SalesMaterial	STANDARD	000-014			0.4313			0		
9/6/2021	SalesMaterial	SalesMaterial	STANDARD	000-014						0.04096734		
8/30/2021	SalesMaterial	SalesMaterial	STANDARD	000-014						0.04096734		
8/28/2021	SalesMaterial	SalesMaterial	STANDARD	000-014						0.04096734		
8/21/2021	SalesMaterial	SalesMaterial	STANDARD	000-014			0.4313			0		
9/5/2021	SalesMaterial	SalesMaterial	STANDARD	000-014						0.04096734		

Di akhir laporan, kelompok umur objek diatur sebagai ALL. Baris ALL berisi total kumulatif, termasuk objek yang lebih kecil dari 128 KB, untuk semua kelompok umur untuk hari itu.

8/24/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0	000-014	
9/3/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0.02426125	015-029	
8/28/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0.03545875	015-029	
8/17/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0	000-014	
8/25/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0	000-014	
9/6/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0.0209529	015-029	
9/4/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0.02304819	015-029	
8/22/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0	000-014	
8/21/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0	000-014	
8/30/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0.03073092	015-029	
8/20/2021	SalesMaterial	SalesMaterial	STANDARD	ALL	3		0.4599			0	000-014	

Bagian selanjutnya menjelaskan kolom yang digunakan dalam laporan.

Tata letak file yang diekspor

Tabel berikut menjelaskan tata letak file yang diekspor.

Mengonfigurasi analisis kelas penyimpanan

Dengan menggunakan alat analisis kelas penyimpanan analitik Amazon S3, Anda dapat menganalisis pola akses penyimpanan untuk membantu memutuskan waktu yang tepat agar dapat memindahkan data yang tepat ke kelas penyimpanan yang tepat. Analisis kelas penyimpanan mengamati pola akses data untuk membantu Anda menentukan waktu untuk memindahkan penyimpanan STANDARD yang jarang diakses ke kelas penyimpanan STANDARD_IA (IA, untuk akses yang jarang). Untuk informasi selengkapnya tentang STANDARD_IA, lihat FAQ Amazon S3 dan Menggunakan kelas penyimpanan Amazon S3.

Anda mengatur analisis kelas penyimpanan dengan mengonfigurasi data objek yang ingin dianalisis. Anda dapat mengonfigurasi analisis kelas penyimpanan untuk melakukan hal berikut:

- Menganalisis seluruh konten bucket.

  Anda akan menerima analisis untuk semua objek dalam bucket.
- Menganalisis objek yang dikelompokkan bersama berdasarkan prefiks dan tanda.

  Anda dapat mengonfigurasi filter yang mengelompokkan objek untuk analisis berdasarkan prefiks, tanda objek, atau kombinasi prefiks dan tanda. Anda akan menerima analisis terpisah untuk setiap filter yang dikonfigurasi. Anda dapat memiliki banyak konfigurasi filter per bucket, hingga 1.000.

- Mengekspor data analisis.

  Saat mengonfigurasi analisis kelas penyimpanan untuk bucket atau filter, Anda dapat memilih untuk mengekspor data analisis ke file setiap hari. Analisis untuk hari itu ditambahkan ke file agar membentuk log analisis historis untuk filter yang dikonfigurasi. File ini diperbarui setiap hari di tujuan pilihan Anda. Saat memilih data yang akan diekspor, Anda menentukan bucket tujuan dan prefiks tujuan opsional tempat file akan ditulis.

Anda dapat menggunakan konsol Amazon S3, REST API, AWS CLI atau AWS SDK untuk mengonfigurasi analisis kelas penyimpanan.

> **Important**
>
> Analisis kelas penyimpanan tidak memberikan rekomendasi untuk pemindahan ke kelas penyimpanan ONEZONE_IA atau S3 Glacier Flexible Retrieval.
> Jika Anda ingin mengonfigurasi analisis kelas penyimpanan untuk mengekspor temuan Anda sebagai file.csv dan bucket tujuan menggunakan enkripsi bucket default dengan a AWS KMS key, Anda harus memperbarui kebijakan AWS KMS kunci untuk memberikan izin Amazon S3 untuk mengenkripsi file.csv. Untuk petunjuk, lihat Memberikan izin kepada Amazon S3 untuk menggunakan CMK untuk enkripsi.

Untuk informasi selengkapnya tentang analitik, lihat Analitik Amazon S3–Analisis Kelas Penyimpanan.

Menggunakan konsol S3

Untuk mengonfigurasi analisis kelas penyimpanan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Dalam daftar Bucket, pilih nama bucket yang ingin Anda konfigurasikan analisis kelas penyimpanannya.
3. Pilih tab Metrik.
4. Di Bawah Analisis Kelas Penyimpanan, pilih Buat konfigurasi analitik.
5. Ketik nama untuk filter. Jika Anda ingin menganalisis seluruh bucket, biarkan kolom Prefiks kosong.

6. Di kolom Prefiks, ketik teks untuk prefiks objek yang ingin Anda analisis.
7. Untuk menambahkan tanda, pilih Tambahkan tanda. Masukkan kunci dan nilai untuk tanda. Anda dapat memasukkan satu prefiks dan beberapa tanda.
8. Atau, Anda dapat memilih Aktifkan di bawah Ekspor CSV untuk mengekspor laporan analisis ke dalam file datar yang nilainya dipisahkan koma (.csv). Pilih bucket tujuan tempat file dapat disimpan. Anda dapat mengetikkan prefiks untuk bucket tujuan. Bucket tujuan harus Wilayah AWS sama dengan ember tempat Anda menyiapkan analisis. Bucket tujuan dapat berada di Akun AWS yang berbeda.

   Jika bucket tujuan untuk file.csv menggunakan enkripsi bucket default dengan kunci KMS, Anda harus memperbarui kebijakan AWS KMS kunci untuk memberikan izin Amazon S3 untuk mengenkripsi file.csv. Untuk petunjuk, lihat Memberikan izin kepada Amazon S3 untuk menggunakan CMK untuk enkripsi.
9. Pilih Buat Konfigurasi.

Amazon S3 membuat kebijakan bucket pada bucket tujuan yang memberikan izin penulisan kepada Amazon S3. Ini akan memungkinkannya untuk menulis data ekspor ke ember.

Apabila terjadi kesalahan saat mencoba membuat kebijakan bucket, Anda akan diberikan petunjuk untuk memperbaikinya. Misalnya, jika Anda memilih bucket tujuan di Akun AWS lain dan tidak memiliki izin untuk membaca dan menulis kebijakan bucket, Anda akan melihat pesan berikut. Anda harus meminta pemilik bucket tujuan menambahkan kebijakan bucket yang ditampilkan ke bucket tujuan. Jika kebijakan tersebut tidak ditambahkan ke bucket tujuan, Anda tidak akan mendapatkan data ekspor karena Amazon S3 tidak memiliki izin untuk menulis ke bucket tujuan. Jika bucket sumber dimiliki oleh akun yang berbeda dengan akun pengguna saat ini, ID akun yang benar dari bucket sumber harus diganti dalam kebijakan.

Untuk informasi tentang data yang diekspor dan cara kerja filter, lihat Analitik Amazon S3–Analisis Kelas Penyimpanan.

Penggunaan API REST

Untuk mengonfigurasi Analisis Kelas Penyimpanan menggunakan REST API, gunakan file PutBucketAnalyticsConfiguration. Anda juga dapat menggunakan operasi yang setara dengan AWS CLI atau AWS SDK.

Anda dapat menggunakan API REST berikut untuk bekerja dengan Analisis Kelas Penyimpanan:

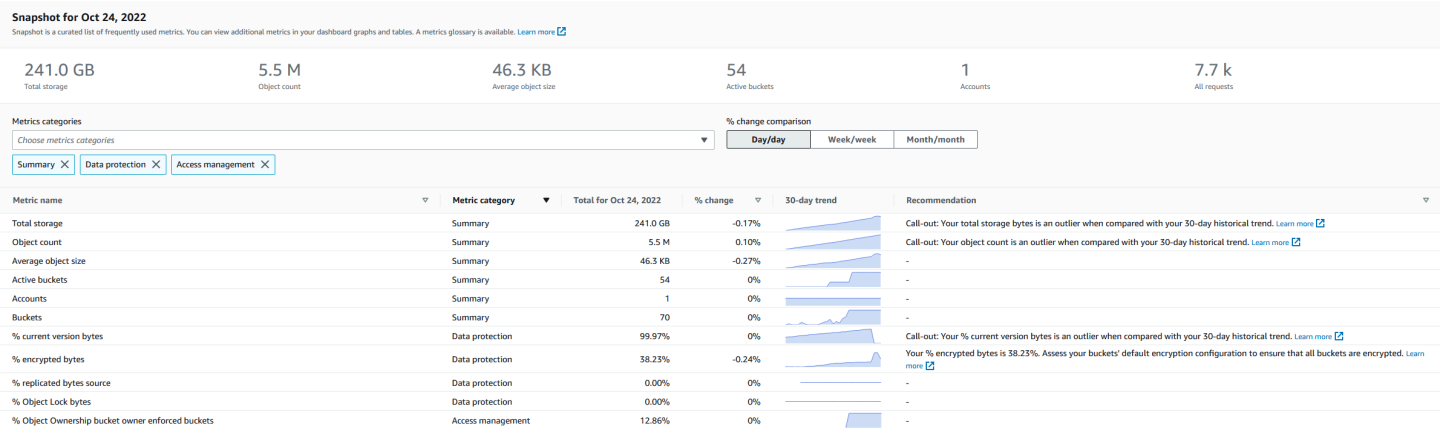
- Konfigurasi DELETE Bucket Analytics
- Kofigurasi GET Bucket Analytics
- Konfigurasi List Bucket Analytics

Menilai aktivitas penyimpanan dan penggunaan Anda dengan Amazon S3 Storage Lens

Amazon S3 Storage Lens adalah fitur analitik penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penyimpanan dan aktivitas objek. S3 Storage Lens juga menganalisis metrik untuk memberikan rekomendasi kontekstual yang dapat Anda gunakan untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik untuk melindungi data Anda.

Anda dapat menggunakan metrik Lensa Penyimpanan S3 untuk menghasilkan wawasan ringkasan. Misalnya, Anda dapat mengetahui berapa banyak penyimpanan yang Anda miliki di seluruh organisasi Anda atau ember dan awalan mana yang paling cepat berkembang. Anda juga dapat menggunakan metrik Lensa Penyimpanan S3 untuk mengidentifikasi peluang pengoptimalan biaya, menerapkan praktik terbaik perlindungan data dan manajemen akses, serta meningkatkan kinerja beban kerja aplikasi. Misalnya, Anda dapat mengidentifikasi bucket yang tidak memiliki aturan Siklus Hidup S3 untuk membatalkan unggahan multibagian yang tidak lengkap yang berusia lebih dari 7 hari. Anda juga dapat mengidentifikasi bucket yang tidak mengikuti praktik terbaik perlindungan data, seperti menggunakan Replikasi S3 atau Pembuatan Versi S3.

Lensa Penyimpanan S3 menggabungkan metrik Anda dan menampilkan informasi di bagian Snapshot akun di halaman Bucket konsol Amazon S3. S3 Storage Lens juga menyediakan dasbor interaktif yang dapat Anda gunakan untuk memvisualisasikan wawasan dan tren, menandai outlier, dan menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik perlindungan data. Dasbor Anda memiliki opsi penelusuran untuk menghasilkan dan memvisualisasikan wawasan di tingkat grup organisasi, akun, kelas penyimpanan, bucketWilayah AWS, awalan, atau Storage Lens. Anda juga dapat mengirim ekspor metrik harian dalam CSV atau Parquet format ke bucket S3.



Metrik dan fitur Lensa Penyimpanan S3

S3 Storage Lens menyediakan dasbor default interaktif yang diperbarui setiap hari. S3 Storage Lens telah mengonfigurasi dasbor ini untuk memvisualisasikan wawasan dan tren yang diringkas untuk seluruh akun Anda dan memperbaruinya setiap hari di konsol S3. Metrik dari dasbor ini juga dirangkum dalam snapshot akun Anda di halaman Bucket. Untuk informasi selengkapnya, lihat Dasbor default.

Untuk membuat dasbor lain dan mencakupnya berdasarkanWilayah AWS, bucket S3, atau akun (untukAWS Organizations), Anda membuat konfigurasi dasbor Lensa Penyimpanan S3. Anda dapat membuat dan mengelola konfigurasi dasbor S3 Storage Lens dengan menggunakan konsol Amazon S3AWS Command Line Interface, AWS CLI (), SDKAWS, atau Amazon S3 REST API. Saat membuat atau mengedit dasbor Lensa Penyimpanan S3, Anda menentukan cakupan dasbor dan pemilihan metrik.

S3 Storage Lens menawarkan metrik gratis dan metrik serta rekomendasi lanjutan, yang dapat Anda tingkatkan dengan biaya tambahan. Dengan metrik dan rekomendasi lanjutan, Anda dapat mengakses metrik dan fitur tambahan untuk mendapatkan wawasan tentang penyimpanan Anda. Fitur-fitur ini mencakup kategori metrik lanjutan, agregasi awalan, rekomendasi kontekstual, dan penerbitan Amazon. CloudWatch Agregasi awalan dan rekomendasi kontekstual hanya tersedia di konsol Amazon S3. Untuk informasi tentang harga S3 Storage Lens, lihat harga Amazon S3.

Kategori metrik

Dalam tingkatan gratis dan lanjutan, metrik diatur ke dalam kategori yang selaras dengan kasus penggunaan utama, seperti pengoptimalan biaya dan perlindungan data. Metrik gratis mencakup ringkasan, pengoptimalan biaya, perlindungan data, manajemen akses, kinerja, dan metrik acara. Saat meningkatkan ke metrik dan rekomendasi lanjutan, Anda dapat mengaktifkan metrik pengoptimalan biaya dan perlindungan data tingkat lanjut. Anda dapat menggunakan metrik canggih

ini untuk lebih mengurangi biaya penyimpanan S3 Anda dan meningkatkan sikap perlindungan data Anda. Anda juga dapat mengaktifkan metrik aktivitas dan metrik kode status terperinci untuk meningkatkan kinerja beban kerja aplikasi yang mengakses bucket S3 Anda. Untuk informasi selengkapnya tentang kategori metrik gratis dan lanjutan, lihatPilihan metrik.

Anda dapat menilai penyimpanan berdasarkan praktik terbaik S3, seperti menganalisis persentase bucket yang memiliki enkripsi atau S3 Object Lock atau S3 Versioning diaktifkan. Anda juga dapat mengidentifikasi peluang penghematan biaya potensial. Misalnya, Anda dapat menggunakan metrik jumlah aturan Siklus Hidup S3 untuk mengidentifikasi bucket yang tidak memiliki aturan kedaluwarsa atau transisi siklus hidup. Anda juga dapat menganalisis aktivitas permintaan per bucket untuk menemukan bucket tempat objek dapat dialihkan ke kelas penyimpanan berbiaya lebih rendah. Untuk informasi selengkapnya, lihat Amazon S3.

Ekspor metrik

Selain melihat dasbor di konsol S3, Anda dapat mengekspor metrik dalam CSV atau Parquet memformat ke bucket S3 untuk analisis lebih lanjut dengan alat analisis pilihan Anda. Untuk informasi selengkapnya, lihat Melihat metrik Lensa Penyimpanan Amazon S3 menggunakan ekspor data.

CloudWatch Penerbitan Amazon

Anda dapat mempublikasikan penggunaan S3 Storage Lens dan metrik aktivitas ke Amazon CloudWatch untuk membuat tampilan terpadu kesehatan operasional Anda di dasbor. CloudWatch Anda juga dapat menggunakan CloudWatch fitur, seperti alarm dan tindakan yang dipicu, matematika metrik, dan deteksi anomali, untuk memantau dan mengambil tindakan pada metrik Lensa Penyimpanan S3. Selain itu, operasi CloudWatch API memungkinkan aplikasi, termasuk penyedia pihak ketiga, untuk mengakses metrik Lensa Penyimpanan S3 Anda. Opsi CloudWatch penerbitan tersedia untuk dasbor yang ditingkatkan ke metrik dan rekomendasi lanjutan S3 Storage Lens. Untuk informasi selengkapnya tentang dukungan untuk metrik Lensa Penyimpanan S3 di CloudWatch, lihat. Pantau metrik S3 Storage Lens di CloudWatch

Untuk informasi selengkapnya tentang penggunaan Lensa Penyimpanan S3, lihat topik berikut.

Topik

- Memahami Amazon S3 Storage Lens
- Menggunakan Amazon S3 Storage Lens dengan AWS Organizations
- Izin Lensa Penyimpanan Amazon S3
- Melihat metrik dengan Lensa Penyimpanan Amazon S3
- Amazon S3

- Glosarium metrik Amazon S3 Storage Lens
- Bekerja dengan Amazon S3 Storage Lens dengan menggunakan konsol dan API
- Bekerja dengan grup Lensa Penyimpanan

Memahami Amazon S3 Storage Lens

> **Important**
>
> Amazon S3 sekarang menerapkan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi untuk setiap bucket di Amazon S3. Mulai 5 Januari 2023, semua unggahan objek baru ke Amazon S3 secara otomatis dienkripsi tanpa biaya tambahan dan tidak berdampak pada kinerja. Status enkripsi otomatis untuk konfigurasi enkripsi default bucket S3 dan untuk unggahan objek baru tersedia di AWS CloudTrail log, S3 Inventory, S3 Storage Lens, konsol Amazon S3, dan sebagai header respons API Amazon S3 tambahan di dan SDK. AWS Command Line Interface AWS Untuk informasi selengkapnya, lihat FAQ enkripsi default.

Amazon S3 Storage Lens adalah fitur analitik penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan objek. Anda dapat menggunakan metrik Lensa Penyimpanan S3 untuk menghasilkan wawasan ringkasan, seperti mencari tahu berapa banyak penyimpanan yang Anda miliki di seluruh organisasi atau bucket dan awalan mana yang paling cepat berkembang. Anda juga dapat menggunakan metrik Lensa Penyimpanan S3 untuk mengidentifikasi peluang pengoptimalan biaya, menerapkan praktik terbaik perlindungan data dan keamanan, serta meningkatkan kinerja beban kerja aplikasi. Misalnya, Anda dapat mengidentifikasi bucket yang tidak memiliki aturan Siklus Hidup S3 untuk mengakhiri unggahan multibagian yang tidak lengkap yang berusia lebih dari 7 hari. Anda juga dapat mengidentifikasi bucket yang tidak mengikuti praktik terbaik perlindungan data, seperti menggunakan Replikasi S3 atau Pembuatan Versi S3. S3 Storage Lens juga menganalisis metrik untuk memberikan rekomendasi kontekstual yang dapat Anda gunakan untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik untuk melindungi data Anda.

Lensa Penyimpanan S3 menggabungkan metrik Anda dan menampilkan informasi di bagian Snapshot akun di halaman Bucket konsol Amazon S3. S3 Storage Lens juga menyediakan dasbor interaktif yang dapat Anda gunakan untuk memvisualisasikan wawasan dan tren, menandai outlier, dan menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik

terbaik perlindungan data. Dasbor Anda memiliki opsi penelusuran untuk menghasilkan dan memvisualisasikan wawasan di tingkat grup organisasi, akun, kelas penyimpanan, bucketWilayah AWS, awalan, atau Storage Lens. Anda juga dapat mengirim ekspor metrik harian dalam CSV atau Parquet format ke bucket S3. Anda dapat membuat dan mengelola dasbor Lensa Penyimpanan S3 menggunakan konsol Amazon S3AWS Command Line Interface, AWS CLI (), SDKAWS, atau Amazon S3 REST API.

Konsep dan terminologi Lensa Penyimpanan S3

Bagian ini berisi terminologi dan konsep yang penting untuk berhasil memahami dan menggunakan Amazon S3 Storage Lens.

Topik

- Konfigurasi dasbor
- Dasbor default
- Dasbor
- Snapshot akun
- Ekspor metrik
- Wilayah Asal
- Periode penahanan
- Kategori metrik
- Rekomendasi
- Pilihan metrik
- S3 Storage Lens dan AWS Organizations

Konfigurasi dasbor

Lensa Penyimpanan S3 memerlukan konfigurasi dasbor yang berisi properti yang diperlukan untuk menggabungkan metrik atas nama Anda untuk satu dasbor atau ekspor. Saat Anda membuat konfigurasi, Anda memilih nama dasbor dan Wilayah beranda, yang tidak dapat Anda ubah setelah membuat dasbor. Anda dapat menambahkan tag secara opsional dan mengonfigurasi ekspor metrik dalam CSV atau format. Parquet

Dalam konfigurasi dasbor, Anda juga menentukan cakupan dasbor dan pemilihan metrik. Cakupan dapat mencakup semua penyimpanan untuk akun organisasi Anda atau bagian yang difilter berdasarkan Wilayah, bucket, dan akun. Saat mengonfigurasi pemilihan metrik, Anda memilih

antara metrik gratis dan metrik dan rekomendasi lanjutan, yang dapat Anda tingkatkan dengan biaya tambahan. Dengan metrik dan rekomendasi lanjutan, Anda dapat mengakses metrik dan fitur tambahan. Fitur-fitur ini mencakup kategori metrik lanjutan, agregasi tingkat awalan, rekomendasi kontekstual, dan penerbitan Amazon. CloudWatch Untuk informasi tentang harga S3 Storage Lens, lihat harga Amazon S3.

Dasbor default

Dasbor default S3 Storage Lens pada konsol diberi nama default-account-dashboard. S3 mengkonfigurasi dasbor ini untuk memvisualisasikan wawasan dan tren yang diringkas untuk seluruh akun Anda dan memperbaruinya setiap hari di konsol S3. Anda tidak dapat mengubah cakupan konfigurasi dasbor default, tetapi Anda dapat meningkatkan pilihan metrik dari metrik gratis ke metrik dan rekomendasi lanjutan. Anda dapat mengonfigurasi ekspor metrik opsional atau bahkan menonaktifkan dasbor. Namun, Anda tidak dapat menghapus dasbor default.

> **Note**
>
> Jika Anda menonaktifkan dasbor default Anda, itu tidak lagi diperbarui. Anda tidak akan lagi menerima metrik harian baru di dasbor Lensa Penyimpanan S3, ekspor metrik, atau snapshot akun di halaman S3 Bucket. Jika dasbor Anda menggunakan metrik dan rekomendasi lanjutan, Anda tidak akan dikenakan biaya lagi. Anda masih dapat melihat data historis di dasbor hingga periode 14 hari untuk kueri data berakhir. Periode ini adalah 15 bulan jika Anda telah mengaktifkan metrik dan rekomendasi lanjutan. Untuk mengakses data historis, Anda dapat mengaktifkan kembali dasbor dalam periode kedaluwarsa.

Dasbor

Anda dapat membuat dasbor Lensa Penyimpanan S3 tambahan dan mencakupnya berdasarkanWilayah AWS, bucket S3, atau akun (untuk). AWS Organizations Saat membuat atau mengedit dasbor Lensa Penyimpanan S3, Anda menentukan cakupan dasbor dan pemilihan metrik. S3 Storage Lens menawarkan metrik gratis dan metrik serta rekomendasi lanjutan, yang dapat Anda tingkatkan dengan biaya tambahan. Dengan metrik dan rekomendasi lanjutan, Anda dapat mengakses metrik dan fitur tambahan untuk mendapatkan wawasan tentang penyimpanan Anda. Ini termasuk kategori metrik lanjutan, agregasi tingkat awalan, rekomendasi kontekstual, dan penerbitan Amazon. CloudWatch Untuk informasi tentang harga S3 Storage Lens, lihat harga Amazon S3.

Anda juga dapat menonaktifkan atau menghapus dasbor. Jika Anda menonaktifkan dasbor, dasbor tidak lagi diperbarui, dan Anda tidak akan lagi menerima metrik harian baru. Anda masih dapat

melihat data historis hingga periode kedaluwarsa 14 hari. Jika Anda mengaktifkan metrik dan rekomendasi lanjutan untuk dasbor tersebut, periode ini adalah 15 bulan. Untuk mengakses data historis, Anda dapat mengaktifkan kembali dasbor dalam periode kedaluwarsa.

Jika Anda menghapus dasbor Anda, Anda akan kehilangan semua pengaturan konfigurasi dasbor Anda. Anda tidak akan lagi menerima metrik harian baru, dan Anda juga kehilangan akses ke data historis yang terkait dengan dasbor tersebut. Jika Anda ingin mengakses data historis untuk dasbor yang telah dihapus, Anda harus membuat dasbor lain dengan nama yang sama di Wilayah asal yang sama.

> **Note**
>
> - Anda dapat menggunakan S3 Storage Lens untuk membuat hingga 50 dasbor per Wilayah asal.
> - Dasbor tingkat organisasi hanya dapat dibatasi untuk lingkup Regional.

Snapshot akun

Snapshot Akun Lensa Penyimpanan S3 merangkum metrik dari dasbor default Anda dan menampilkan total penyimpanan, jumlah objek, dan ukuran objek rata-rata di halaman Bucket konsol S3. Snapshot akun ini memberi Anda akses cepat ke wawasan tentang penyimpanan Anda tanpa harus meninggalkan halaman Bucket. Snapshot akun juga menyediakan akses sekali klik ke dasbor Lensa Penyimpanan S3 interaktif Anda.

Anda dapat menggunakan dasbor untuk memvisualisasikan wawasan dan tren, menandai outlier, dan menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik perlindungan data. Dasbor Anda memiliki opsi penelusuran untuk menghasilkan wawasan di tingkat organisasi, akun, bucket, objek, atau awalan. Anda juga dapat mengirim ekspor metrik sekali sehari ke bucket S3 dalam CSV atau format. Parquet

Anda tidak dapat mengubah cakupan dasbor dasbor akun default karena ditautkan ke snapshot Akun. Namun, Anda dapat meningkatkan pilihan metrik default-account-dashboarddari metrik gratis ke metrik dan rekomendasi lanjutan berbayar. Setelah memutakhirkan, Anda kemudian dapat menampilkan semua permintaan, byte yang diunggah, dan byte yang diunduh di snapshot Akun Lensa Penyimpanan S3.

> **Note**
>
> Jika Anda menonaktifkan dasbor default, snapshot Akun Anda tidak lagi diperbarui. Untuk terus menampilkan metrik di snapshot Akun, Anda dapat mengaktifkan kembali metrik. default-account-dashboard

Ekspor metrik

Ekspor metrik S3 Storage Lens adalah file yang berisi semua metrik yang diidentifikasi dalam konfigurasi S3 Storage Lens Anda. Informasi ini dihasilkan setiap hari dalam CSV atau Parquet format dan dikirim ke bucket S3. Anda dapat menggunakan ekspor metrik untuk analisis lebih lanjut dengan menggunakan alat metrik pilihan Anda. Bucket S3 untuk ekspor metrik harus berada dalam Wilayah yang sama dengan konfigurasi S3 Storage Lens Anda. Anda dapat menghasilkan ekspor metrik Lensa Penyimpanan S3 dari konsol S3 dengan mengedit konfigurasi dasbor Anda. Anda juga dapat mengonfigurasi ekspor metrik dengan menggunakan AWS CLI dan AWS SDK.

Wilayah Asal

Wilayah rumah adalah Wilayah AWS tempat semua metrik Lensa Penyimpanan S3 untuk konfigurasi dasbor tertentu disimpan. Anda harus memilih Wilayah rumah saat membuat konfigurasi dasbor Lensa Penyimpanan S3 Anda. Setelah Anda memilih Wilayah rumah, Anda tidak dapat mengubahnya. Selain itu, jika Anda membuat grup Lensa Penyimpanan, kami sarankan Anda memilih Wilayah rumah yang sama dengan dasbor Lensa Penyimpanan Anda.

> **Note**
>
> Anda dapat memilih salah satu Wilayah berikut sebagai wilayah asal Anda:
>
> - AS Timur (Virginia Utara) – `us-east-1`
> - AS Timur (Ohio) – `us-east-2`
> - AS Barat (N. California) – `us-west-1`
> - AS Barat (Oregon) – `us-west-2`
> - Asia Pasifik (Mumbai) – `ap-south-1`
> - Asia Pasifik (Seoul) – `ap-northeast-2`
> - Asia Pasifik (Singapura) – `ap-southeast-1`
> - Asia Pasifik (Sydney) – `ap-southeast-2`

- Asia Pasifik (Tokyo) – `ap-northeast-1`
- Kanada (Pusat) – `ca-central-1`
- Tiongkok (Beijing) — `cn-north-1`
- Tiongkok (Ningxia) — `cn-northwest-1`
- Eropa (Frankfurt) – `eu-central-1`
- Eropa (Irlandia) – `eu-west-1`
- Eropa (London) – `eu-west-2`
- Eropa (Paris) – `eu-west-3`
- Eropa (Stockholm) – `eu-north-1`
- Amerika Selatan (São Paulo) – `sa-east-1`

Periode penahanan

Metrik Lensa Penyimpanan S3 dipertahankan sehingga Anda dapat melihat tren historis dan membandingkan perbedaan dalam penyimpanan dan aktivitas Anda dari waktu ke waktu. Anda dapat menggunakan metrik Lensa Penyimpanan Amazon S3 untuk kueri sehingga Anda dapat melihat tren historis dan membandingkan perbedaan dalam penggunaan dan aktivitas penyimpanan Anda dari waktu ke waktu.

Semua metrik Lensa Penyimpanan S3 dipertahankan untuk jangka waktu 15 bulan. Namun, metrik hanya tersedia untuk kueri selama durasi tertentu, yang bergantung pada pemilihan metrik Anda. Durasi ini tidak dapat diubah. Metrik gratis tersedia untuk kueri selama periode 14 hari, dan metrik lanjutan tersedia untuk kueri selama periode 15 bulan.

Kategori metrik

Dalam tingkatan gratis dan lanjutan, metrik Lensa Penyimpanan S3 disusun ke dalam kategori yang selaras dengan kasus penggunaan utama, seperti pengoptimalan biaya dan perlindungan data. Metrik gratis mencakup ringkasan, pengoptimalan biaya, perlindungan data, manajemen akses, kinerja, dan metrik acara. Saat meningkatkan ke metrik dan rekomendasi lanjutan, Anda dapat mengaktifkan metrik pengoptimalan biaya dan perlindungan data tambahan yang dapat Anda gunakan untuk lebih mengurangi biaya penyimpanan S3 dan memastikan data Anda terlindungi. Anda juga dapat mengaktifkan metrik aktivitas dan metrik kode status terperinci yang dapat Anda gunakan untuk meningkatkan kinerja alur kerja aplikasi.

Daftar berikut menunjukkan semua kategori metrik gratis dan lanjutan. Untuk daftar lengkap metrik individual yang disertakan dalam setiap kategori, lihat Glosarium metrik.

Metrik ringkasan

Metrik ringkasan memberikan wawasan umum tentang penyimpanan S3 Anda, termasuk total byte penyimpanan dan jumlah objek.

Metrik pengoptimalan biaya

Metrik pengoptimalan biaya memberikan wawasan yang dapat Anda gunakan untuk mengelola dan mengoptimalkan biaya penyimpanan Anda. Misalnya, Anda dapat mengidentifikasi bucket yang memiliki unggahan multipart yang tidak lengkap yang berusia lebih dari 7 hari.

Dengan metrik dan rekomendasi lanjutan, Anda dapat mengaktifkan metrik pengoptimalan biaya tingkat lanjut. Metrik ini mencakup metrik jumlah aturan Siklus Hidup S3 yang dapat Anda gunakan untuk mendapatkan kedaluwarsa per bucket dan jumlah aturan Siklus Hidup transisi S3.

Metrik perlindungan data

Metrik perlindungan data memberikan wawasan untuk fitur perlindungan data, seperti enkripsi dan Pembuatan Versi S3. Anda dapat menggunakan metrik ini untuk mengidentifikasi bucket yang tidak mengikuti praktik terbaik perlindungan data. Misalnya, Anda dapat mengidentifikasi bucket yang tidak menggunakan enkripsi default dengan AWS Key Management Service kunci (SSE-KMS) atau S3 Versioning.

Dengan metrik dan rekomendasi lanjutan, Anda dapat mengaktifkan metrik perlindungan data tingkat lanjut. Metrik ini mencakup metrik jumlah aturan replikasi per ember.

Metrik manajemen akses

Metrik manajemen akses memberikan wawasan untuk Kepemilikan Objek S3. Anda dapat menggunakan metrik ini untuk melihat setelan Kepemilikan Objek yang digunakan bucket Anda.

Metrik acara

Metrik acara memberikan wawasan untuk Pemberitahuan Acara S3. Dengan metrik acara, Anda dapat melihat bucket mana yang memiliki S3 Event Notifications yang dikonfigurasi.

Metrik kinerja

Metrik kinerja memberikan wawasan untuk Akselerasi Transfer S3. Dengan metrik kinerja, Anda dapat melihat bucket mana yang mengaktifkan Akselerasi Transfer.

Metrik aktivitas (lanjutan)

Jika Anda memutakhirkan dasbor ke Metrik dan rekomendasi lanjutan, Anda dapat mengaktifkan metrik aktivitas. Metrik aktivitas memberikan detail tentang cara penyimpanan Anda diminta (misalnya, Semua permintaan, Dapatkan permintaan, Masukkan permintaan), Byte yang diunggah atau diunduh, dan kesalahan.

Metrik aktivitas tingkat awalan dapat digunakan untuk membantu Anda menentukan awalan mana yang jarang digunakan, sehingga Anda dapat beralih ke kelas penyimpanan yang lebih optimal menggunakan Siklus Hidup S3.

Metrik kode status terperinci (lanjutan)

Jika Anda memutakhirkan dasbor ke metrik dan rekomendasi lanjutan, Anda dapat mengaktifkan metrik kode status terperinci. Metrik kode status terperinci memberikan wawasan untuk kode status HTTP, seperti 403 Forbidden dan 503 Service Unavailable, yang dapat Anda gunakan untuk memecahkan masalah akses atau kinerja. Misalnya, Anda dapat melihat metrik jumlah kesalahan 403 Forbidden untuk mengidentifikasi beban kerja yang mengakses bucket tanpa izin yang benar diterapkan.

Metrik kode status terperinci tingkat awalan dapat digunakan untuk mendapatkan pemahaman yang lebih baik tentang kemunculan kode status HTTP dengan awalan. Misalnya, 503 metrik jumlah kesalahan memungkinkan Anda mengidentifikasi awalan yang menerima permintaan pembatasan selama penyerapan data.

Rekomendasi

S3 Storage Lens memberikan rekomendasi otomatis untuk membantu Anda mengoptimalkan penyimpanan Anda. Rekomendasi ditempatkan secara kontekstual bersama metrik yang relevan di dasbor S3 Storage Lens. Data historis tidak memenuhi syarat untuk rekomendasi karena rekomendasi relevan dengan apa yang terjadi pada periode terbaru. Rekomendasi hanya muncul jika relevan.

Rekomendasi S3 Storage Lens tersedia dalam bentuk berikut:

- Saran

  Saran mengingatkan Anda tentang tren dalam penyimpanan dan aktivitas Anda yang mungkin menunjukkan peluang pengoptimalan biaya penyimpanan atau praktik terbaik perlindungan data. Anda dapat menggunakan topik yang disarankan di Panduan Pengguna Amazon S3 dan dasbor

Lensa Penyimpanan S3 untuk menelusuri detail selengkapnya tentang Wilayah, bucket, atau awalan tertentu.

- Panggilan

  Call-out adalah rekomendasi yang mengingatkan Anda akan anomali menarik dalam penyimpanan dan aktivitas Anda selama periode yang mungkin memerlukan perhatian atau pemantauan lebih lanjut.

  - Panggilan outlier

    S3 Storage Lens menyediakan panggilan keluar untuk metrik yang outlier, berdasarkan tren 30 hari Anda baru-baru ini. Outlier dihitung dengan menggunakan skor standar, juga dikenal sebagai skor-z. Dalam skor ini, metrik hari ini dikurangi dari rata-rata 30 hari terakhir untuk metrik tersebut. Metrik hari ini kemudian dibagi dengan standar deviasi untuk metrik tersebut selama 30 hari terakhir. Skor yang dihasilkan biasanya antara -3 dan +3. Angka ini mewakili jumlah standar deviasi bahwa metrik hari ini berasal dari mean.

    S3 Storage Lens menganggap metrik dengan skor >2 atau <-2 sebagai outlier karena lebih tinggi atau lebih rendah dari 95 persen data terdistribusi normal.

  - Perubahan panggilan yang signifikan

    Perubahan yang signifikan berlaku untuk metrik yang diharapkan lebih jarang berubah. Oleh karena itu, ini diatur ke sensitivitas yang lebih tinggi daripada perhitungan outlier, yang biasanya dalam kisaran +/- 20 persen versus hari, minggu, atau bulan sebelumnya.

    Mengatasi panggilan keluar di penyimpanan dan aktivitas Anda — Jika Anda menerima panggilan keluar perubahan yang signifikan, itu belum tentu menjadi masalah. Panggilan bisa menjadi hasil dari perubahan yang diantisipasi dalam penyimpanan Anda. Misalnya, Anda mungkin baru saja menambahkan sejumlah besar objek baru, menghapus sejumlah besar objek, atau membuat perubahan terencana yang serupa.

    Jika Anda melihat call-out perubahan signifikan di dasbor Anda, perhatikan dan tentukan apakah itu dapat dijelaskan oleh keadaan baru-baru ini. Jika tidak, gunakan dasbor S3 Storage Lens untuk menelusuri detail selengkapnya guna memahami Wilayah, bucket, atau prefiks tertentu yang mendorong fluktuasi.

- Pengingat

Pengingat memberikan wawasan tentang cara kerja Amazon S3. Mereka dapat membantu Anda mempelajari lebih lanjut tentang cara menggunakan fitur S3 untuk mengurangi biaya penyimpanan atau menerapkan praktik terbaik perlindungan data.

Pilihan metrik

S3 Storage Lens menawarkan dua pilihan metrik yang dapat Anda pilih untuk dasbor dan ekspor Anda:metrik gratis dan metrik dan rekomendasi tingkat lanjut.

- Metrik gratis

  S3 Storage Lens menawarkan metrik gratis untuk semua dasbor dan konfigurasi. Metrik gratis berisi metrik yang relevan dengan penyimpanan Anda, seperti jumlah bucket dan objek di akun Anda. Metrik gratis juga menyertakan metrik berbasis kasus penggunaan (misalnya, pengoptimalan biaya dan metrik perlindungan data) yang dapat Anda gunakan untuk menyelidiki apakah penyimpanan Anda dikonfigurasi sesuai dengan praktik terbaik S3. Semua metrik gratis dikumpulkan setiap hari. Data tersedia untuk kueri selama 14 hari. Untuk informasi selengkapnya tentang metrik mana yang tersedia dengan metrik gratis, lihat. Glosarium metrik Amazon S3 Storage Lens

- Metrik dan rekomendasi lanjutan

  S3 Storage Lens menawarkan metrik gratis untuk semua dasbor dan konfigurasi dengan opsi untuk meningkatkan ke metrik dan rekomendasi lanjutan. Biaya tambahan berlaku. Untuk informasi selengkapnya, lihat Harga Amazon S3.

  Metrik dan rekomendasi lanjutan mencakup semua metrik dalam metrik gratis bersama dengan metrik tambahan, seperti metrik perlindungan data dan pengoptimalan biaya tingkat lanjut, metrik aktivitas, dan metrik kode status terperinci. Metrik dan rekomendasi lanjutan juga memberikan rekomendasi untuk membantu Anda mengoptimalkan penyimpanan Anda. Rekomendasi ditempatkan secara kontekstual bersama metrik yang relevan di dasbor.

  Metrik dan rekomendasi lanjutan mencakup fitur-fitur berikut:

  - Metrik lanjutan — Hasilkan metrik tambahan. Untuk daftar lengkap kategori metrik lanjutan, lihatKategori metrik. Untuk daftar metrik lengkap, lihat. Glosarium metrik Amazon S3 Storage Lens
  - Amazon CloudWatch publishing — Menerbitkan metrik Lensa Penyimpanan S3 CloudWatch untuk membuat tampilan terpadu kesehatan operasional Anda di dasbor. CloudWatch Anda juga

dapat menggunakan operasi dan fitur CloudWatch API, seperti alarm dan tindakan yang dipicu, matematika metrik, dan deteksi anomali, untuk memantau dan mengambil tindakan pada metrik Lensa Penyimpanan S3. Untuk informasi selengkapnya, lihat Pantau metrik S3 Storage Lens di CloudWatch.

- Agregasi awalan - Mengumpulkan metrik di tingkat awalan. Mengaktifkan agregasi awalan memperluas semua metrik yang disertakan dalam konfigurasi dasbor Anda di tingkat awalan. Metrik hanya dibuat untuk awalan yang memenuhi ambang batas yang dikonfigurasi. Perhatikan bahwa metrik yang berlaku pada tingkat awalan tersedia dengan agregasi Awalan, kecuali untuk pengaturan tingkat ember dan metrik jumlah aturan. Metrik tingkat awalan tidak dipublikasikan ke. CloudWatch
- Agregasi grup Lensa Penyimpanan — Mengumpulkan metrik di tingkat grup Lensa Penyimpanan. Setelah mengaktifkan metrik dan rekomendasi lanjutan serta agregasi grup Lensa Penyimpanan, Anda dapat menentukan grup Lensa Penyimpanan mana yang akan disertakan atau dikecualikan dari dasbor Lensa Penyimpanan. Setidaknya satu grup Lensa Penyimpanan harus ditentukan. Grup Lensa Penyimpanan yang ditentukan juga harus berada di dalam Wilayah rumah yang ditunjuk di akun dasbor. Metrik tingkat grup Lensa Penyimpanan tidak dipublikasikan ke. CloudWatch

Semua metrik lanjutan dikumpulkan setiap hari. Data tersedia untuk kueri hingga 15 bulan. Untuk informasi selengkapnya tentang metrik penyimpanan yang dikumpulkan oleh S3 Storage Lens, lihat. Glosarium metrik Amazon S3 Storage Lens

> **Note**
>
> Rekomendasi hanya tersedia saat Anda menggunakan dasbor Lensa Penyimpanan S3 di konsol Amazon S3.

S3 Storage Lens dan AWS Organizations

AWS Organizationsadalah Layanan AWS yang membantu Anda menggabungkan semua hierarki Akun AWS di bawah satu organisasi Anda. Amazon S3 Storage Lens berfungsi AWS Organizations untuk menyediakan satu tampilan penyimpanan objek dan aktivitas di seluruh penyimpanan Amazon S3 Anda.

Untuk informasi selengkapnya, lihat Menggunakan Amazon S3 Storage Lens dengan AWS Organizations.

- Akses tepercaya

  Menggunakan akun manajemen organisasi Anda, Anda harus mengaktifkan akses tepercaya untuk S3 Storage Lens untuk menggabungkan metrik penyimpanan dan data penggunaan untuk semua akun anggota di organisasi Anda. Anda kemudian dapat membuat dasbor atau ekspor untuk organisasi Anda dengan menggunakan akun manajemen Anda atau dengan memberikan akses administrator yang didelegasikan ke akun lain di organisasi Anda.

  Anda dapat menonaktifkan akses tepercaya untuk S3 Storage Lens kapan saja, yang menghentikan S3 Storage Lens dari menggabungkan metrik untuk organisasi Anda.

- Administrator yang didelegasikan

  Anda dapat membuat dasbor dan metrik untuk S3 Storage Lens untuk organisasi Anda dengan menggunakan akun AWS Organizations manajemen Anda, atau dengan memberikan akses administrator yang didelegasikan ke akun lain di organisasi Anda. Anda dapat membatalkan pendaftaran administrator yang didelegasikan kapan saja. Membatalkan pendaftaran administrator yang didelegasikan juga secara otomatis menghentikan semua dasbor tingkat organisasi yang dibuat oleh administrator yang didelegasikan tersebut untuk menggabungkan metrik penyimpanan baru.

Untuk informasi lebih lanjut, lihat Amazon S3 Storage Lens and AWS Organizations di Panduan Pengguna AWS Organizations.

Peran terkait layanan Amazon S3 Storage Lens

Bersama dengan akses tepercaya AWS Organizations, Amazon S3 Storage Lens menggunakan peran terkait layanan AWS Identity and Access Management (IAM). Peran terkait layanan adalah tipe IAM role unik yang tertaut langsung ke Lensa Penyimpanan S3. Peran terkait layanan telah ditentukan sebelumnya oleh S3 Storage Lens dan menyertakan semua izin yang diperlukan untuk mengumpulkan penyimpanan harian dan metrik aktivitas dari akun anggota di organisasi Anda.

Untuk informasi selengkapnya, lihat Menggunakan Peran Terkait Layanan untuk Amazon S3 Storage Lens.

Menggunakan Amazon S3 Storage Lens dengan AWS Organizations

Amazon S3 Storage Lens adalah fitur analitik penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan

objek. Anda dapat menggunakan metrik Lensa Penyimpanan S3 untuk menghasilkan wawasan ringkasan, seperti mencari tahu berapa banyak penyimpanan yang Anda miliki di seluruh organisasi atau bucket dan awalan mana yang paling cepat berkembang. Anda juga dapat menggunakan metrik Lensa Penyimpanan S3 untuk mengidentifikasi peluang pengoptimalan biaya, menerapkan praktik terbaik perlindungan data dan keamanan, serta meningkatkan kinerja beban kerja aplikasi. Misalnya, Anda dapat mengidentifikasi bucket yang tidak memiliki aturan Siklus Hidup S3 untuk mengakhiri unggahan multibagian yang tidak lengkap yang berusia lebih dari 7 hari. Anda juga dapat mengidentifikasi bucket yang tidak mengikuti praktik terbaik perlindungan data, seperti menggunakan Replikasi S3 atau Pembuatan Versi S3. S3 Storage Lens juga menganalisis metrik untuk memberikan rekomendasi kontekstual yang dapat Anda gunakan untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik untuk melindungi data Anda.

Anda dapat menggunakan Amazon S3 Storage Lens untuk mengumpulkan metrik penyimpanan dan data penggunaan untuk semua Akun AWS yang merupakan bagian dari hierarki AWS Organizations Anda. Untuk melakukan ini, Anda harus menggunakanAWS Organizations, dan Anda harus mengaktifkan akses tepercaya S3 Storage Lens dengan menggunakan akun AWS Organizations manajemen Anda.

Setelah mengaktifkan akses tepercaya, Anda dapat menambahkan akses administrator yang didelegasikan ke akun di organisasi Anda. Akun ini kemudian dapat membuat konfigurasi S3 Storage Lens dan dasbor yang mengumpulkan metrik penyimpanan dan data pengguna di seluruh organisasi.

Untuk informasi lebih lanjut dalam mengaktifkan akses tepercaya, lihat Amazon S3 Storage Lens and AWS Organizations di Panduan Pengguna AWS Organizations.

Topik

- Mengaktifkan akses tepercaya untuk S3 Storage Lens
- Menonaktifkan akses tepercaya untuk S3 Storage Lens
- Mendaftarkan administrator yang didelegasikan untuk S3 Storage Lens
- Membatalkan pendaftaran administrator yang didelegasikan untuk S3 Storage Lens

Mengaktifkan akses tepercaya untuk S3 Storage Lens

Dengan mengaktifkan akses tepercaya, Anda mengizinkan Lensa Penyimpanan Amazon S3 memiliki akses ke hierarki, keanggotaan, dan struktur AWS Organizations AWS Organizations Anda melalui operasi API. S3 Storage Lens kemudian menjadi layanan tepercaya untuk seluruh struktur organisasi Anda.

Setiap kali konfigurasi dasbor dibuat, S3 Storage Lens membuat peran terkait layanan di manajemen organisasi atau akun administrator yang didelegasikan. Peran terkait layanan memberikan izin S3 Storage Lens untuk melakukan hal berikut:

- Jelaskan organisasi
- Daftar akun
- Verifikasi daftar Layanan AWS akses untuk organisasi
- Dapatkan administrator yang didelegasikan untuk organisasi

S3 Storage Lens kemudian dapat memastikan bahwa ia memiliki akses untuk mengumpulkan metrik lintas akun untuk akun di organisasi Anda. Untuk informasi selengkapnya, lihat Menggunakan Peran Terkait Layanan untuk Amazon S3 Storage Lens.

Setelah mengaktifkan akses tepercaya, Anda dapat menetapkan akses administrator yang didelegasikan ke akun di organisasi Anda. Bila akun ditandai sebagai administrator yang didelegasikan untuk suatu layanan, akun akan menerima otorisasi untuk mengakses semua operasi API organisasi hanya-baca. Akses ini memberikan visibilitas administrator yang didelegasikan ke anggota dan struktur organisasi Anda sehingga mereka juga dapat membuat dasbor Lensa Penyimpanan S3.

> **Note**
>
> Hanya akun manajemen yang dapat mengaktifkan akses tepercaya untuk Amazon S3 Storage Lens.

Menonaktifkan akses tepercaya untuk S3 Storage Lens

Dengan menonaktifkan akses tepercaya, Anda membatasi S3 Storage Lens untuk bekerja hanya pada tingkat akun. Selain itu, setiap pemegang akun hanya dapat melihat informasi Lensa Penyimpanan S3 untuk ruang lingkup akun mereka, dan bukan seluruh organisasi mereka. Dasbor apa pun yang memerlukan akses tepercaya tidak lagi diperbarui, tetapi akan menyimpan data historisnya selama periode data tersedia untuk kueri.

> **Note**
>
> - Menonaktifkan akses tepercaya untuk S3 Storage Lens juga secara otomatis menghentikan semua dasbor tingkat organisasi untuk mengumpulkan dan menggabungkan metrik penyimpanan.
> - Akun administrator manajemen dan delegasi Anda akan tetap dapat melihat data historis untuk dasbor tingkat organisasi yang ada selama periode data tersedia untuk kueri.

Mendaftarkan administrator yang didelegasikan untuk S3 Storage Lens

Anda dapat membuat dasbor tingkat organisasi menggunakan akun manajemen organisasi atau akun administrator yang didelegasikan. Akun administrator yang didelegasikan mengizinkan akun lain selain akun manajemen Anda untuk membuat dasbor tingkat organisasi. Hanya akun manajemen organisasi yang dapat mendaftarkan dan membatalkan pendaftaran akun lain sebagai administrator yang didelegasikan untuk organisasi.

Untuk mendaftarkan administrator yang didelegasikan menggunakan konsol Amazon S3, lihat. Mendaftarkan administrator yang didelegasikan untuk Lensa Penyimpanan S3

Anda juga dapat mendaftarkan administrator yang didelegasikan menggunakan AWS Organizations REST APIAWS CLI, atau SDK dari akun manajemen. Untuk informasi selengkapnya, lihat RegisterDelegatedAdministrator dalam Referensi API AWS Organizations.

> **Note**
>
> Sebelum Anda dapat menunjuk administrator yang didelegasikan dengan menggunakan AWS Organizations REST API,AWS CLI, atau SDK, Anda harus memanggil operasi. EnableAWSOrganizationsAccess

Membatalkan pendaftaran administrator yang didelegasikan untuk S3 Storage Lens

Anda juga dapat membatalkan pendaftaran akun administrator yang didelegasikan. Akun administrator yang didelegasikan mengizinkan akun lain selain akun manajemen Anda untuk membuat dasbor tingkat organisasi. Hanya akun manajemen organisasi yang dapat membatalkan pendaftaran akun sebagai administrator yang didelegasikan untuk organisasi.

Untuk membatalkan pendaftaran administrator yang didelegasikan menggunakan konsol S3, lihat. Membatalkan pendaftaran administrator yang didelegasikan untuk Lensa Penyimpanan S3

Anda juga dapat membatalkan pendaftaran administrator yang didelegasikan dengan menggunakan AWS Organizations REST APIAWS CLI, atau SDK dari akun manajemen. Untuk informasi selengkapnya, lihat DeregisterDelegatedAdministrator dalam Referensi API AWS Organizations.

> **Note**
>
> - Membatalkan pendaftaran administrator yang didelegasikan juga secara otomatis menghentikan semua dasbor tingkat organisasi yang dibuat oleh administrator yang didelegasikan tersebut untuk menggabungkan metrik penyimpanan baru.
> - Administrator delegasi yang dideregistrasi masih dapat melihat data historis untuk dasbor yang mereka buat saat data tersedia untuk kueri.

Izin Lensa Penyimpanan Amazon S3

Amazon S3 Storage Lens memerlukan izin baru di AWS Identity and Access Management (IAM) untuk mengotorisasi akses ke tindakan S3 Storage Lens. Untuk memberikan izin ini, Anda dapat menggunakan kebijakan IAM berbasis identitas. Anda dapat melampirkan kebijakan ini ke pengguna, grup, atau peran IAM untuk memberi mereka izin. Izin tersebut dapat mencakup kemampuan untuk mengaktifkan atau menonaktifkan Lensa Penyimpanan S3, atau untuk mengakses dasbor atau konfigurasi Lensa Penyimpanan S3 apa pun.

Pengguna atau peran IAM harus menjadi milik akun yang membuat atau memiliki dasbor atau konfigurasi, kecuali kedua kondisi berikut benar:

- Akun Anda adalah anggotaAWS Organizations.
- Anda diberi akses untuk membuat dasbor tingkat organisasi oleh akun manajemen Anda sebagai administrator yang didelegasikan.

> **Note**
>
> - Anda tidak dapat menggunakan kredensi pengguna root akun Anda untuk melihat dasbor Lensa Penyimpanan Amazon S3. Untuk mengakses dasbor Lensa Penyimpanan S3, Anda

harus memberikan izin IAM yang diperlukan kepada pengguna IAM baru atau yang sudah ada. Kemudian, masuk dengan kredensial pengguna tersebut untuk mengakses dasbor S3 Storage Lens. Untuk informasi selengkapnya tentang administrator, lihat Praktik Terbaik IAM dalam Panduan Pengguna IAM.

- Menggunakan S3 Storage Lens pada konsol Amazon S3 dapat memerlukan beberapa izin. Misalnya, untuk mengedit dasbor di konsol, Anda memerlukan izin berikut:
  - `s3:ListStorageLensConfigurations`
  - `s3:GetStorageLensConfiguration`
  - `s3:PutStorageLensConfiguration`

Topik

- Mengatur izin akun untuk menggunakan S3 Storage Lens
- Menyetel izin akun untuk menggunakan grup Lensa Penyimpanan S3
- Mengatur izin untuk menggunakan S3 Storage Lens dengan AWS Organizations

Mengatur izin akun untuk menggunakan S3 Storage Lens

Untuk membuat dan mengelola dasbor Lensa Penyimpanan S3 dan konfigurasi dasbor Lensa Penyimpanan, Anda harus memiliki izin berikut, tergantung pada tindakan yang ingin Anda lakukan:

Izin IAM terkait Amazon S3 Storage Lens

Tindakan	Izin IAM

Buat atau perbarui dasbor S3 Storage Lens di konsol Amazon S3.	`s3:ListStorageLensConfigurations`
	`s3:GetStorageLensConfiguration`
	`s3:GetStorageLensConfigurationTagging`
	`s3:PutStorageLensConfiguration`
	`s3:PutStorageLensConfigurationTagging`

Tindakan	Izin IAM
Dapatkan tag dasbor Lensa Penyimpanan S3 di konsol Amazon S3.	`s3:ListStorageLensConfigurations` `s3:GetStorageLensConfigurationTagging`

Melihat dasbor S3 Storage Lens di konsol Amazon S3.	`s3:ListStorageLensConfigurations`
	`s3:GetStorageLensConfiguration`
	`s3:GetStorageLensDashboard`

Menghapus dasbor S3 Storage Lens di konsol Amazon S3.	`s3:ListStorageLensConfigurations`
	`s3:GetStorageLensConfiguration`
	`s3:DeleteStorageLensConfiguration`

Membuat atau memperbarui konfigurasi S3 Storage Lens dengan menggunakan AWS CLI atau AWS SDK.	`s3:PutStorageLensConfiguration` `s3:PutStorageLensConfigurationTagging`
Dapatkan tag konfigurasi Lensa Penyimpanan S3 dengan menggunakan AWS CLI atau AWS SDK.	`s3:GetStorageLensConfigurationTagging`
Melihat konfigurasi Lensa Penyimpanan S3 menggunakan AWS CLI atau AWS SDK.	`s3:GetStorageLensConfiguration`
Hapus konfigurasi Lensa Penyimpanan S3 dengan menggunakan AWS CLI atau AWS SDK.	`s3:DeleteStorageLensConfiguration`

Note

- Anda dapat menggunakan tag sumber daya dalam kebijakan IAM untuk mengelola izin.

- Pengguna IAM atau peran dengan izin ini dapat melihat metrik dari bucket dan awalan yang mungkin tidak memiliki izin langsung untuk membaca atau mencantumkan objek.
- Untuk dasbor Lensa Penyimpanan S3 dengan metrik tingkat awalan diaktifkan, jika jalur awalan yang dipilih cocok dengan kunci objek, dasbor mungkin menampilkan kunci objek sebagai awalan lain.
- Untuk ekspor metrik, yang disimpan dalam bucket di akun Anda, izin diberikan dengan menggunakan izin yang ada dalam kebijakan `s3:GetObject` IAM. Demikian pula, untuk AWS Organizations entitas, akun manajemen organisasi atau akun administrator yang didelegasikan dapat menggunakan kebijakan IAM untuk mengelola izin akses untuk dasbor dan konfigurasi tingkat organisasi.

Menyetel izin akun untuk menggunakan grup Lensa Penyimpanan S3

Anda dapat menggunakan grup Lensa Penyimpanan S3 untuk memahami distribusi penyimpanan Anda dalam bucket berdasarkan awalan, akhiran, tag objek, ukuran objek, atau usia objek. Anda dapat melampirkan grup Lensa Penyimpanan ke dasbor untuk melihat metrik agregatnya.

Untuk bekerja dengan grup Lensa Penyimpanan, Anda memerlukan izin tertentu. Untuk informasi selengkapnya, lihat the section called “Izin grup Lensa Penyimpanan”.

Mengatur izin untuk menggunakan S3 Storage Lens dengan AWS Organizations

Anda dapat menggunakan Amazon S3 Storage Lens untuk mengumpulkan metrik penyimpanan dan data penggunaan untuk semua akun yang merupakan bagian dari hierarki AWS Organizations Anda. Berikut ini adalah tindakan dan izin yang terkait dengan menggunakan S3 Storage Lens dengan Organizations.

Izin IAM terkait AWS Organizations untuk menggunakan S3 Storage Lens

Tindakan	Izin IAM
Aktifkan akses tepercaya untuk S3 Storage Lens untuk organisasi Anda.	`organizations:EnableAWSServiceAccess`
Nonaktifkan akses tepercaya untuk S3 Storage Lens untuk organisasi Anda.	`organizations:DisableAWSServiceAccess`

Tindakan	Izin IAM
Daftarkan administrator yang didelegasikan untuk membuat dasbor atau konfigurasi S3 Storage Lens untuk organisasi Anda.	`organizations:RegisterDelegatedAdministrator`
Batalkan pendaftaran administrator yang didelegasikan sehingga mereka tidak dapat lagi membuat dasbor atau konfigurasi Lensa Penyimpanan S3 untuk organisasi Anda.	`organizations:DeregisterDelegatedAdministrator`

Izin tambahan untuk membuat konfigurasi seluruh organisasi Lensa Penyimpanan S3.	`organizations:DescribeOrganization`
	
	`organizations:ListAccounts`
	
	`organizations:ListAWSServiceAccessForOrganization`
	
	`organizations:ListDelegatedAdministrators`
	
	`iam:CreateServiceLinkedRole`

Melihat metrik dengan Lensa Penyimpanan Amazon S3

Lensa Penyimpanan S3 menggabungkan metrik Anda dan menampilkan informasi di bagian Snapshot akun di halaman Bucket konsol Amazon S3. S3 Storage Lens juga menyediakan dasbor interaktif yang dapat Anda gunakan untuk memvisualisasikan wawasan dan tren, menandai outlier, dan menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik perlindungan data. Dasbor Anda memiliki opsi penelusuran untuk menghasilkan dan memvisualisasikan wawasan di tingkat grup organisasi, akun, kelas penyimpanan, bucketWilayah AWS, awalan, atau Storage Lens. Anda juga dapat mengirim ekspor metrik harian dalam CSV atau Parquet format ke bucket S3.

Secara default, semua dasbor dikonfigurasi dengan metrik gratis, yang mencakup metrik yang dapat Anda gunakan untuk memahami penggunaan dan aktivitas di seluruh penyimpanan S3 Anda, mengoptimalkan biaya penyimpanan Anda, dan menerapkan praktik terbaik perlindungan data dan

manajemen akses. Metrik gratis digabungkan ke tingkat bucket. Dengan metrik gratis, data tersedia untuk kueri hingga 14 hari.

Metrik dan rekomendasi lanjutan mencakup fitur tambahan berikut yang dapat Anda gunakan untuk mendapatkan wawasan lebih lanjut tentang penggunaan dan aktivitas di seluruh penyimpanan Anda dan praktik terbaik untuk mengoptimalkan penyimpanan Anda:

- Rekomendasi kontekstual (hanya tersedia di dasbor)
- Metrik lanjutan (termasuk metrik aktivitas yang dikumpulkan berdasarkan bucket)
- Agregasi awalan
- Agregasi grup Lensa Penyimpanan
- Agregasi grup Lensa Penyimpanan
- CloudWatch Penerbitan Amazon

Data metrik lanjutan tersedia untuk kueri selama 15 bulan. Ada biaya tambahan untuk menggunakan S3 Storage Lens dengan metrik tingkat lanjut. Untuk informasi selengkapnya, lihat Harga Amazon S3. Untuk informasi selengkapnya tentang metrik gratis dan lanjutan, lihatPilihan metrik.

Topik

- Melihat metrik S3 Storage Lens di dasbor
- Melihat metrik Lensa Penyimpanan Amazon S3 menggunakan ekspor data
- Pantau metrik S3 Storage Lens di CloudWatch

Melihat metrik S3 Storage Lens di dasbor

Di konsol Amazon S3, S3 Storage Lens menyediakan dasbor default interaktif yang dapat Anda gunakan untuk memvisualisasikan wawasan dan tren dalam data Anda. Anda juga dapat menggunakan dasbor ini untuk menandai outlier dan menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik perlindungan data. Dasbor Anda memiliki opsi penelusuran untuk menghasilkan wawasan di tingkat grup akun, bucket, awalanWilayah AWS, atau Storage Lens. Jika Anda telah mengaktifkan S3 Storage Lens untuk bekerja denganAWS Organizations, Anda juga dapat menghasilkan wawasan di tingkat organisasi (seperti data untuk semua akun yang merupakan bagian dari AWS Organizations hierarki Anda). Dasbor selalu dimuat untuk tanggal terbaru yang memiliki metrik yang tersedia.

Dasbor default S3 Storage Lens pada konsol diberi nama default-account-dashboard. Amazon S3 melakukan pra-konfigurasi dasbor ini untuk memvisualisasikan wawasan dan tren yang diringkas untuk seluruh akun Anda dan memperbaruinya setiap hari di konsol S3. Anda tidak dapat mengubah cakupan konfigurasi dasbor default, tetapi Anda dapat meningkatkan pilihan metrik dari metrik gratis ke metrik dan rekomendasi lanjutan berbayar. Dengan metrik dan rekomendasi lanjutan, Anda dapat mengakses metrik dan fitur tambahan. Fitur-fitur ini mencakup kategori metrik lanjutan, agregasi tingkat awalan, rekomendasi kontekstual, dan penerbitan Amazon. CloudWatch

Anda dapat menonaktifkan dasbor default, tetapi Anda tidak dapat menghapusnya. Jika Anda menonaktifkan dasbor default Anda, itu tidak lagi diperbarui. Anda juga tidak akan lagi menerima metrik harian baru di Lensa Penyimpanan S3 atau di bagian snapshot Akun di halaman Bucket. Anda masih dapat melihat data historis di dasbor default hingga periode 14 hari untuk kueri data berakhir. Periode ini adalah 15 bulan jika Anda telah mengaktifkan metrik dan rekomendasi lanjutan. Untuk mengakses data ini, Anda dapat mengaktifkan kembali dasbor default dalam periode kedaluwarsa.

Anda dapat membuat dasbor Lensa Penyimpanan S3 tambahan dan mencakupnya berdasarkanWilayah AWS, bucket S3, atau akun. Anda juga dapat membuat cakupan dasbor berdasarkan organisasi jika Anda telah mengaktifkan Storage Lens untuk bekerja dengannyaAWS Organizations. Saat membuat atau mengedit dasbor Lensa Penyimpanan S3, Anda menentukan cakupan dasbor dan pemilihan metrik.

Anda dapat menonaktifkan atau menghapus dasbor tambahan apa pun yang Anda buat.

- Jika Anda menonaktifkan dasbor, dasbor tidak lagi diperbarui, dan Anda tidak akan lagi menerima metrik harian baru. Anda masih dapat melihat data historis untuk metrik gratis hingga periode kedaluwarsa 14 hari. Jika Anda mengaktifkan metrik dan rekomendasi lanjutan untuk dasbor tersebut, periode ini adalah 15 bulan. Untuk mengakses data ini, Anda dapat mengaktifkan kembali dasbor dalam periode kedaluwarsa.
- Jika Anda menghapus dasbor Anda, Anda akan kehilangan semua pengaturan konfigurasi dasbor Anda. Anda tidak akan lagi menerima metrik harian baru, dan Anda juga kehilangan akses ke data historis yang terkait dengan dasbor tersebut. Jika Anda ingin mengakses data historis untuk dasbor yang telah dihapus, Anda harus membuat dasbor lain dengan nama yang sama di Wilayah asal yang sama.

Topik

- Melihat dasbor Amazon S3 Storage Lens
- Memahami dasbor S3 Storage Lens Anda

Melihat dasbor Amazon S3 Storage Lens

Prosedur berikut menunjukkan cara melihat dasbor Lensa Penyimpanan S3 di konsol S3. Untuk panduan berbasis kasus penggunaan yang menunjukkan cara menggunakan dasbor Anda untuk mengoptimalkan biaya, menerapkan praktik terbaik, dan meningkatkan kinerja aplikasi yang mengakses bucket S3 Anda, lihat. Amazon S3

> **Note**
>
> Anda tidak dapat menggunakan kredensi pengguna root akun Anda untuk melihat dasbor Lensa Penyimpanan Amazon S3. Untuk mengakses dasbor Lensa Penyimpanan S3, Anda harus memberikan izin yang diperlukan AWS Identity and Access Management (IAM) kepada pengguna IAM baru atau yang sudah ada. Kemudian, masuk dengan kredensial pengguna tersebut untuk mengakses dasbor S3 Storage Lens. Untuk informasi selengkapnya, lihat Izin Lensa Penyimpanan Amazon S3 dan Praktik terbaik keamanan di IAM di Panduan Pengguna IAM.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Lensa Penyimpanan, Dasbor.
3. Di daftar Dasbor, pilih dasbor yang ingin Anda lihat.

   Dasbor Anda terbuka di Lensa Penyimpanan S3. Bagian Snapshot untuk tanggal menunjukkan tanggal terbaru yang telah dikumpulkan oleh S3 Storage Lens. Dasbor Anda selalu memuat tanggal terbaru yang memiliki metrik yang tersedia.
4. (Opsional) Untuk mengubah tanggal dasbor Lensa Penyimpanan S3 Anda, di pemilih tanggal kanan atas, pilih tanggal baru.
5. (Opsional) Untuk menerapkan filter sementara untuk lebih membatasi cakupan data dasbor Anda, lakukan hal berikut:

   a. Perluas bagian Filter.

   b. Untuk memfilter berdasarkan akun tertentuWilayah AWS, kelas penyimpanan, bucket, awalan, atau grup Lensa Penyimpanan, pilih opsi yang akan difilter.

> **Note**
>
> Filter Awalan dan filter grup Lensa Penyimpanan tidak dapat diterapkan secara bersamaan.

c. Untuk memperbarui filter, pilih Terapkan.

d. Untuk menghapus filter, klik pada X di sebelah filter.

6. Di bagian mana pun di dasbor Lensa Penyimpanan S3 Anda, untuk melihat data untuk metrik tertentu, untuk Metrik, pilih nama metrik.
7. Dalam bagan atau visualisasi apa pun di dasbor Lensa Penyimpanan S3, Anda dapat menelusuri tingkat agregasi yang lebih dalam dengan menggunakan tab Grup Akun,, Kelas Penyimpanan Wilayah AWS, Bucket, Awalan, atau Lensa Penyimpanan. Sebagai contoh, lihat Temukan ember Amazon S3 yang dingin.

Memahami dasbor S3 Storage Lens Anda

Dasbor Lensa Penyimpanan S3 Anda memiliki tab Ikhtisar utama, dan hingga lima tab tambahan yang mewakili setiap tingkat agregasi:

- Akun
- Wilayah AWS
- Kelas penyimpanan
- Ember
- Awalan
- Grup Lensa Penyimpanan

Pada tab Ikhtisar, data dasbor Anda digabungkan menjadi tiga bagian berbeda: Snapshot untuk tanggal, Tren dan distribusi, dan ikhtisar N Teratas.

Untuk informasi selengkapnya tentang dasbor Lensa Penyimpanan S3 Anda, lihat bagian berikut.

Snapshot

Bagian Snapshot untuk tanggal menunjukkan metrik ringkasan yang telah dikumpulkan oleh Lensa Penyimpanan S3 untuk tanggal yang dipilih. Metrik ringkasan ini mencakup metrik berikut:

- Total penyimpanan — Jumlah total penyimpanan yang digunakan dalam byte.
- Jumlah objek — Jumlah total objek dalam AndaAkun AWS.
- Ukuran objek rata-rata — Ukuran objek rata-rata.
- Bucket aktif — Jumlah total bucket aktif dalam penggunaan aktif dengan penyimpanan > 0 byte di akun Anda.
- Akun — Jumlah akun yang penyimpanannya berada dalam ruang lingkup. Nilai ini adalah 1 kecuali Anda menggunakan AWS Organizations dan Lensa Penyimpanan S3 Anda memiliki akses tepercaya dengan peran terkait layanan yang valid. Untuk informasi selengkapnya, lihat Menggunakan Peran Terkait Layanan untuk Lensa Penyimpanan Amazon S3.
- Bucket — Jumlah total bucket di akun Anda.

Data metrik

Untuk setiap metrik yang muncul di snapshot, Anda dapat melihat data berikut:

- Nama metrik — Nama metrik.
- Kategori metrik — Kategori tempat metrik disusun.
- Total untuk tanggal - Jumlah total untuk tanggal yang dipilih.
- % perubahan — Persentase perubahan dari tanggal snapshot terakhir.
- Tren 30 hari — Garis tren yang menunjukkan perubahan metrik selama periode 30 hari.
- Rekomendasi — Rekomendasi kontekstual berdasarkan data yang disediakan dalam snapshot. Rekomendasi tersedia dengan metrik dan rekomendasi lanjutan. Untuk informasi selengkapnya, lihat Rekomendasi.

Kategori metrik

Anda dapat secara opsional memperbarui Snapshot dasbor Anda untuk bagian tanggal untuk menampilkan metrik untuk kategori lain. Jika ingin melihat data snapshot untuk metrik tambahan, Anda dapat memilih dari kategori Metrik berikut:

- Optimalisasi biaya
- Perlindungan data
- Aktivitas (tersedia dengan metrik lanjutan)
- Manajemen akses

- Kinerja
- Acara

Bagian Snapshot untuk tanggal hanya menampilkan pilihan metrik untuk setiap kategori. Untuk melihat semua metrik untuk kategori tertentu, pilih metrik di bagian Tren dan distribusi atau ikhtisar N Teratas. Untuk informasi selengkapnya tentang kategori metrik, lihatKategori metrik. Untuk daftar lengkap metrik Lensa Penyimpanan S3, lihat. Glosarium metrik Amazon S3 Storage Lens

Tren dan distribusi

Bagian kedua dari tab Ikhtisar adalah Tren dan distribusi. Di bagian Tren dan distribusi, Anda dapat memilih dua metrik untuk membandingkan rentang tanggal yang Anda tentukan. Bagian Tren dan distribusi menunjukkan hubungan antara dua metrik dari waktu ke waktu. Bagian ini menampilkan bagan yang dapat Anda gunakan untuk melihat kelas Penyimpanan dan distribusi Wilayah antara dua tren yang Anda lacak. Anda dapat secara opsional menelusuri titik data di salah satu grafik untuk analisis yang lebih dalam.

Untuk panduan yang menggunakan bagian Tren dan distribusi, lihat. Identifikasi bucket yang tidak menggunakan enkripsi sisi server denganAWS KMSuntuk enkripsi default (SSE-KMS)

Ikhtisar N teratas

Bagian ketiga dari dasbor S3 Storage Lens adalah Ikhtisar N teratas (diurutkan dalam urutan menaik atau menurun). Bagian ini menampilkan metrik yang Anda pilih di sejumlah besar akun, bucketWilayah AWS, awalan, atau grup Lensa Penyimpanan. Jika Anda mengaktifkan S3 Storage Lens untuk bekerja denganAWS Organizations, Anda juga dapat melihat metrik yang dipilih di seluruh organisasi Anda.

Untuk panduan yang menggunakan bagian ikhtisar N Teratas, lihat. Identifikasi bucket S3 terbesar Anda

Telusuri dan analisis berdasarkan opsi

Untuk memberikan pengalaman analisis yang lancar, dasbor S3 Storage Lens menyediakan menu tindakan, yang muncul saat Anda memilih nilai bagan apa pun. Untuk menggunakan menu ini, pilih nilai bagan apa pun untuk melihat nilai metrik terkait, lalu pilih dari dua opsi di kotak yang muncul:

- Tindakan Drill down menerapkan nilai yang dipilih sebagai filter di semua tab dasbor Anda. Anda kemudian dapat menelusuri ke dalam nilai tersebut untuk analisis mendalam.

- Analisis dengan tindakan akan membawa Anda ke tab Dimensi yang Anda pilih dan menerapkan nilai tab tersebut sebagai filter. Tab ini mencakup Akun,, kelas Penyimpanan Wilayah AWS, Bucket, Prefiks (untuk dasbor yang mengaktifkan metrik Lanjutan dan agregasi Awalan), dan grup Lensa Penyimpanan (untuk dasbor yang mengaktifkan metrik Lanjutan dan agregasi grup Lensa Penyimpanan). Dengan Analyze by, Anda dapat melihat data dalam konteks dimensi baru untuk analisis yang lebih dalam.

Penelusuran dan Analisis berdasarkan tindakan mungkin dinonaktifkan jika hasilnya akan menghasilkan hasil yang tidak logis atau tidak memiliki nilai apa pun. Baik Drill down dan Analyze by actions menerapkan filter di atas semua filter yang ada di semua tab dasbor. Anda juga dapat menghapus filter sesuai kebutuhan.

Tab

Tab tingkat dimensi memberikan tampilan rinci dari semua nilai dalam dimensi tertentu. Misalnya, Wilayah AWStab menampilkan metrik untuk semuaWilayah AWS, dan tab Bucket menampilkan metrik untuk semua bucket. Setiap tab dimensi berisi tata letak identik yang terdiri dari empat bagian:

- Bagan tren yang menampilkan item N teratas Anda dalam dimensi selama 30 hari terakhir untuk metrik yang dipilih. Secara default, bagan ini menampilkan 10 item teratas, tetapi Anda dapat menguranginya menjadi setidaknya 3 item atau meningkatkannya hingga 50 item.
- Bagan histogram yang menunjukkan bagan batang vertikal untuk tanggal dan metrik yang dipilih. Jika Anda memiliki sejumlah besar item untuk ditampilkan dalam bagan ini, Anda mungkin perlu menggulir secara horizontal.
- Bagan analisis gelembung yang memplot semua item dalam dimensi. Bagan ini mewakili metrik pertama pada sumbu x dan metrik kedua pada sumbu y. Metrik ketiga diwakili oleh ukuran gelembung.
- Tampilan kisi metrik yang berisi setiap item dalam dimensi yang tercantum dalam baris. Kolom mewakili setiap metrik yang tersedia, disusun dalam tab kategori metrik untuk navigasi yang lebih mudah.

Melihat metrik Lensa Penyimpanan Amazon S3 menggunakan ekspor data

Metrik Lensa Penyimpanan Amazon S3 dihasilkan setiap hari dalam file ekspor metrik berformat CSV atau Apache Parquet dan ditempatkan dalam bucket S3 di akun Anda. Dari sana, Anda dapat memasukkan ekspor metrik ke alat analisis pilihan Anda, seperti Amazon QuickSight dan Amazon Athena, tempat Anda dapat menganalisis penggunaan penyimpanan dan tren aktivitas.

Topik

- Menggunakan AWS KMS key untuk mengenkripsi ekspor metrik Anda
- Apa yang dimaksud dengan manifes ekspor Lensa Penyimpanan S3?
- Memahami skema ekspor Lensa Penyimpanan Amazon S3

Menggunakan AWS KMS key untuk mengenkripsi ekspor metrik Anda

Untuk memberikan izin kepada Lensa Penyimpanan Amazon S3 guna mengenkripsi ekspor metrik menggunakan CMK, Anda harus menggunakan kebijakan kunci. Untuk memperbarui kebijakan kunci sehingga Anda dapat menggunakan kunci KMS untuk mengenkripsi ekspor metrik Lensa Penyimpanan S3 Anda, ikuti langkah-langkah berikut.

Untuk memberikan izin Lensa Penyimpanan S3 guna mengenkripsi data dengan menggunakan kunci KMS

1. Masuk ke AWS Management Console dengan menggunakan Akun AWS yang memiliki kunci yang dikelola pelanggan.
2. Buka AWS KMS konsol di https://console.aws.amazon.com/kms.
3. Untuk mengubah Wilayah AWS, gunakan pemilih Wilayah di sudut kanan atas halaman.
4. Di panel navigasi kiri, pilih CMK.
5. Di bawah Kunci terkelola pelanggan, pilih kunci yang ingin Anda gunakan untuk mengenkripsi ekspor metrik. AWS KMS keys khusus Wilayah dan harus berada di Wilayah yang sama dengan bucket S3 tujuan ekspor metrik.
6. Di Bawah Kebijakan kunci, pilih Beralih ke tampilan kebijakan.
7. Untuk memperbarui kebijakan kunci, pilih Edit.
8. Di Bawah Edit kebijakan kunci, tambahkan kebijakan kunci berikut ke kebijakan kunci yang ada. Untuk menggunakan kebijakan ini, ganti *user input placeholders* dengan informasi Anda.

```
{
    "Sid": "Allow Amazon S3 Storage Lens use of the KMS key",
     "Effect": "Allow",
    "Principal": {
        "Service": "storage-lens.s3.amazonaws.com"
    },
    "Action": [
        "kms:GenerateDataKey"
    ],
```

```
    "Resource": "*",
    "Condition": {
       "StringEquals": {
           "aws:SourceArn": "arn:aws:s3:us-east-1:source-account-id:storage-
lens/your-dashboard-name",
           "aws:SourceAccount": "source-account-id"
        }
     }
}
```

9. Pilih Simpan perubahan.

Untuk informasi selengkapnya tentang cara membuat CMK, dan menggunakan kebijakan kunci, lihat topik berikut di Panduan Developer AWS Key Management Service :

- Memulai
- Menggunakan kebijakan utama di AWS KMS

Anda juga dapat menggunakan AWS KMS PUT key policy API operation (PutKeyPolicy) untuk menyalin kebijakan kunci ke kunci terkelola pelanggan yang ingin Anda gunakan untuk mengenkripsi ekspor metrik menggunakan REST API AWS CLI, dan SDK.

Apa yang dimaksud dengan manifes ekspor Lensa Penyimpanan S3?

Mengingat banyaknya jumlah data yang dikumpulkan, ekspor metrik harian Lensa Penyimpanan S3 dapat dibagi menjadi beberapa file. File manifes `manifest.json` menjelaskan lokasi file ekspor metrik untuk hari tersebut. Setiap kali ada ekspor baru yang dikirimkan, ekspor tersebut disertai dengan manifes baru. Tiap manifes yang terkandung dalam file `manifest.json` menyediakan metadata dan informasi dasar lainnya tentang ekspor.

Informasi manifes mencakup properti berikut:

- `sourceAccountId`–ID akun dari pemilik konfigurasi.
- `configId`–Pengidentifikasi unik untuk dasbor.
- `destinationBucket`–Bucket tujuan Amazon Resource Name (ARN) tempat metrik ekspor ditempatkan.
- `reportVersion`–Versi dari ekspor.
- `reportDate`–Tanggal dari laporan.

- `reportFormat`–Format dari laporan.
- `reportSchema`–Skema dari laporan.
- `reportFiles`–Daftar aktual dari file laporan ekspor yang berada di bucket tujuan.

Berikut ini adalah contoh manifes dalam file `manifest.json` untuk ekspor berformat CSV.

```
{
   "sourceAccountId":"123456789012",
   "configId":"my-dashboard-configuration-id",
   "destinationBucket":"arn:aws:s3:::destination-bucket",
   "reportVersion":"V_1",
   "reportDate":"2020-11-03",
   "reportFormat":"CSV",

 "reportSchema":"version_number,configuration_id,report_date,aws_account_number,aws_region,stor
   "reportFiles":[
        {
            "key":"DestinationPrefix/StorageLens/123456789012/my-dashboard-
configuration-id/V_1/reports/dt=2020-11-03/a38f6bc4-2e3d-4355-ac8a-e2fdcf3de158.csv",
            "size":1603959,
            "md5Checksum":"2177e775870def72b8d84febe1ad3574"
        }
      ]
}
```

Berikut ini adalah contoh manifes dalam file `manifest.json` untuk ekspor berformat Parquet.

```
{
   "sourceAccountId":"123456789012",
   "configId":"my-dashboard-configuration-id",
   "destinationBucket":"arn:aws:s3:::destination-bucket",
   "reportVersion":"V_1",
   "reportDate":"2020-11-03",
   "reportFormat":"Parquet",
   "reportSchema":"message s3.storage.lens { required string version_number;
 required string configuration_id; required string report_date; required string
 aws_account_number; required string aws_region; required string storage_class;
 required string record_type; required string record_value; required string
 bucket_name; required string metric_name; required long metric_value; }",
   "reportFiles":[
```

```
      {
         "key":"DestinationPrefix/StorageLens/123456789012/my-dashboard-configuration-
id/V_1/reports/dt=2020-11-03/bd23de7c-b46a-4cf4-bcc5-b21aac5be0f5.par",
         "size":14714,
         "md5Checksum":"b5c741ee0251cd99b90b3e8eff50b944"
      }
}
```

Anda dapat mengonfigurasi ekspor metrik agar dibuat sebagai bagian dari konfigurasi dasbor di konsol Amazon S3 atau dengan menggunakan Amazon S3 REST API AWS CLI,, dan SDK.

Memahami skema ekspor Lensa Penyimpanan Amazon S3

Tabel berikut berisi skema ekspor metrik Lensa Penyimpanan S3 Anda.

Nama atribut	Tipe data	Nama kolom	Deskripsi
`VersionNumber`	String	`version_number`	Versi metrik Lensa Penyimpanan S3 yang digunakan.
`ConfigurationId`	String	`configuration_id`	`configuration_id` dari konfigurasi Lensa Penyimpanan S3 Anda.
`ReportDate`	String	`report_date`	Tanggal metrik dilacak.
`AwsAccountNumber`	String	`aws_account_number`	Akun AWS Nomor Anda.
`AwsRegion`	String	`aws_region`	Yang Wilayah AWS metriknya dilacak.
`StorageClass`	String	`storage_class`	Kelas penyimpanan bucket yang dimaksud.

Nama atribut	Tipe data	Nama kolom	Deskripsi
`RecordType`	ENUM	`record_type`	Jenis artefak yang sedang dilaporkan (ACCOUNT, BUCKET, atau PREFIX).
`RecordValue`	String	`record_value`	Nilai artefak `RecordType` . **Note** `record_value` dienkodekan dengan URL.
`BucketName`	String	`bucket_name`	Nama bucket yang sedang dilaporkan.
`MetricName`	String	`metric_name`	Nama metrik yang sedang dilaporkan.
`MetricValue`	Long	`metric_value`	Nilai metrik yang sedang dilaporkan.

Contoh ekspor metrik Lensa Penyimpanan S3

Berikut ini adalah contoh ekspor metrik Lensa Penyimpanan S3 berdasarkan skema ini.

> **Note**
>
> Anda dapat mengidentifikasi metrik untuk Grup Lensa Penyimpanan dengan mencari nilai `STORAGE_LENS_GROUP_BUCKET` atau `STORAGE_LENS_GROUP_ACCOUNT` di kolom `record_type`. Kolom `record_value` akan menampilkan Amazon

Resource Name (ARN) untuk Grup Lensa Penyimpanan, misalnya, `arn:aws:s3:us-east-1:123456789012:storage-lens-group/slg-1`.

version_r	configuration_id	report_date	aws_account_number	aws_region	storage_class	record_type	record_value	bucket_name	metric_name	metric_value
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			StorageBytes	2478830621
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			ObjectCount	1598962
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			ReplicatedStorageBytes	20000
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			ReplicatedObjectCount	20
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			EncryptedStorageBytes	2478828742
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			EncryptedObjectCount	1598961
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			DeleteMarkerObjectCount	1500
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			ObjectLockEnabledStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			ObjectLockEnabledObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			CurrentVersionStorageBytes	2478830621
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			CurrentVersionObjectCount	1598962
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			NonCurrentVersionStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			NonCurrentVersionObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			IncompleteMultipartUploadStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	eu-west-1	STANDARD	ACCOUNT			IncompleteMultipartUploadObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	StorageBytes	29996800
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	ObjectCount	12264
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	ReplicatedStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	ReplicatedObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	EncryptedStorageBytes	29996800
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	EncryptedObjectCount	12264
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	DeleteMarkerObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	ObjectLockEnabledStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	ObjectLockEnabledObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	CurrentVersionStorageBytes	29996800
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	CurrentVersionObjectCount	12264
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	NonCurrentVersionStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	NonCurrentVersionObjectCount	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	IncompleteMultipartUploadStorageBytes	0
V_1	sample-cmh-exclude	11/3/2020	546264889236	us-west-1	STANDARD	PREFIX	AWSLogs%2F546264889236%2FCloudTrail%2Fus	cloudtrail-log-sfo	IncompleteMultipartUploadObjectCount	0

Berikut ini adalah contoh ekspor metrik Lensa Penyimpanan S3 dengan data Grup Lensa Penyimpanan.

version_numbe	configuration_id	report_date	aws_account_num	aws_region	storage_class	record_type	record_value	bucket_name	metric_name	metric_value
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		StorageBytes	3128548856
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ObjectCount	4440
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ReplicatedStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ReplicatedObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		EncryptedStorageBytes	3128548856
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		EncryptedObjectCount	4440
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		DeleteMarkerObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ObjectLockEnabledStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ObjectLockEnabledObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		CurrentVersionStorageBytes	3128548856
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		CurrentVersionObjectCount	4440
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		NonCurrentVersionStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		NonCurrentVersionObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		IncompleteMultipartUploadStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		IncompleteMultipartUploadObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		DeleteMarkerStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ReplicatedStorageBytesSource	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		ReplicatedObjectCountSource	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		IncompleteMPUStorageBytesOlderThan7Days	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_ACCOUNT	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180		IncompleteMPUObjectCountOlderThan7Days	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	StorageBytes	676863200
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ObjectCount	3000
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ReplicatedStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ReplicatedObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	EncryptedStorageBytes	676863200
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	EncryptedObjectCount	3000
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	DeleteMarkerObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ObjectLockEnabledStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ObjectLockEnabledObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	CurrentVersionStorageBytes	676863200
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	CurrentVersionObjectCount	3000
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	NonCurrentVersionStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	NonCurrentVersionObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	IncompleteMultipartUploadStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	IncompleteMultipartUploadObjectCount	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	DeleteMarkerStorageBytes	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ReplicatedStorageBytesSource	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	ReplicatedObjectCountSource	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	IncompleteMPUStorageBytesOlderThan7Days	0
V_1	sample-sfo-exclude	11/15/23	546264889236	us-west-1	STANDARD	STORAGE_LENS_GROUP_BUCKET	arn:aws:s3:us-west-1:546264889236:storage-lens-group/age-more-than-180	cloud-trail-log-sfo	IncompleteMPUObjectCountOlderThan7Days	0

Pantau metrik S3 Storage Lens di CloudWatch

Anda dapat mempublikasikan metrik S3 Storage Lens CloudWatch ke Amazon untuk membuat tampilan terpadu kesehatan operasional Anda di CloudWatchdasbor. Anda juga dapat menggunakan

CloudWatch fitur, seperti alarm dan tindakan yang dipicu, matematika metrik, dan deteksi anomali, untuk memantau dan mengambil tindakan pada metrik Lensa Penyimpanan S3. Selain itu, operasi CloudWatch API memungkinkan aplikasi, termasuk penyedia pihak ketiga, untuk mengakses metrik Lensa Penyimpanan S3 Anda. Untuk informasi selengkapnya tentang CloudWatch fitur, lihat Panduan CloudWatch Pengguna Amazon.

Anda dapat mengaktifkan opsi CloudWatch penerbitan untuk konfigurasi dasbor baru atau yang sudah ada menggunakan konsol Amazon S3, API REST Amazon S3,AWS CLI,AWS SDKs. Dasbor yang ditingkatkan ke metrik dan rekomendasi lanjutan Lensa Penyimpanan S3 dapat menggunakan opsi CloudWatch penerbitan. Untuk harga metrik dan rekomendasi tingkat lanjut S3 Storage Lens, lihat harga Amazon S3. Tidak ada biaya penerbitan CloudWatch metrik tambahan yang berlaku; namun, CloudWatch biaya lain, seperti dasbor, alarm, dan panggilan API, berlaku. Untuk informasi lebih lanjut, lihat harga Amazon CloudWatch.

Metrik S3 Storage Lens dipublikasikan CloudWatch di akun yang memiliki konfigurasi S3 Storage Lens. Setelah Anda mengaktifkan opsi CloudWatch penerbitan dalam metrik dan rekomendasi lanjutan, Anda dapat mengakses metrik tingkat organisasi, akun, dan bucket CloudWatch. Metrik tingkat awalan tidak tersedia di CloudWatch.

> **Note**
>
> Metrik Lensa Penyimpanan S3 adalah metrik harian dan dipublikasikan ke CloudWatch sekali per hari. Saat Anda melakukan kueri pada metrik Lensa Penyimpanan S3 CloudWatch, periode kueri harus 1 hari (86400 detik). Setelah metrik S3 Storage Lens harian Anda muncul di dasbor S3 Storage Lens di konsol Amazon S3, diperlukan beberapa jam agar metrik yang sama ini muncul CloudWatch. Saat Anda mengaktifkan opsi CloudWatch penerbitan untuk metrik Lensa Penyimpanan S3 untuk pertama kalinya, dibutuhkan waktu hingga 24 jam untuk dipublikasikan metrik Anda CloudWatch.

Setelah mengaktifkan opsi CloudWatch penerbitan, Anda dapat menggunakan CloudWatch fitur berikut untuk memantau dan menganalisis data LensStorage Lensa Penyimpanan S3 Anda:

- Dasbor untuk membuat dasbor untuk membuat CloudWatch dasbor untuk membuat dasbor S3 Storage Lens yang disesuaikan. Bagikan CloudWatch dasbor Anda dengan orang-orang yang tidak memiliki akses langsung ke AndaAkun AWS, lintas tim, dan dengan orang-orang di luar organisasi Anda.

- Alarm dan tindakan yang dipicu — Konfigurasi alarm yang menonton metrik dan lakukan tindakan saat ada pelanggaran. Misalnya, Anda dapat mengonfigurasi alarm yang mengirimkan notifikasi Amazon SNS saat metrik Uncomplete Multipart Upload Bytes melebihi 1 GB selama tiga hari berturut-turut.
- Deteksi anomali - Aktifkan deteksi anomali untuk terus menerus menganalisis metrik, menentukan garis dasar normal, dan anomali permukaan. Anda dapat membuat alarm deteksi anomali berdasarkan nilai metrik yang diharapkan. Misalnya, Anda dapat memantau anomali untuk metrik Object Lock Enabled Bytes untuk mendeteksi penghapusan pengaturan Object Lock yang tidak sah.
- Matematika metrik — Anda juga dapat menggunakan matematika metrik untuk membuat kueri beberapa metrik Lensa Penyimpanan S3 dan menggunakan ekspresi matematika untuk membuat deret waktu baru berdasarkan metrik ini. Misalnya, Anda dapat membuat metrik baru untuk mendapatkan ukuran objek rata-rata`StorageBytes` dengan membaginya`ObjectCount`.

Untuk informasi selengkapnya tentang opsi CloudWatch penerbitan untuk metrik S3 Storage Lens, lihat topik-topik berikut.

Topik

- Metrik dan dimensi S3 Storage Lens
- Mengaktifkan CloudWatch penerbitan untuk S3 Storage Lens
- Bekerja dengan metrik S3 Storage Lens di CloudWatch

Metrik dan dimensi S3 Storage Lens

Untuk mengirim metrik Lensa Penyimpanan S3 CloudWatch, Anda harus mengaktifkan opsi CloudWatch penerbitan dalam metrik dan rekomendasi lanjutan S3 Storage Lens. Setelah metrik lanjutan diaktifkan, Anda dapat menggunakan CloudWatchdasbor untuk memantau metrik Lensa Penyimpanan S3 bersama metrik aplikasi lain dan membuat tampilan terpadu tentang kesehatan operasional Anda. Anda dapat menggunakan dimensi untuk memfilter metrik Lensa Penyimpanan S3 CloudWatch menurut organisasi, akun, bucket, kelas penyimpanan, Wilayah, dan ID konfigurasi metrik.

Metrik S3 Storage Lens dipublikasikan CloudWatch di akun yang memiliki konfigurasi S3 Storage Lens. Setelah Anda mengaktifkan opsi CloudWatch penerbitan dalam metrik dan rekomendasi lanjutan, Anda dapat mengakses metrik tingkat organisasi, akun, dan bucket CloudWatch. Metrik tingkat awalan tidak tersedia di CloudWatch.

> **Note**
>
> Metrik Lensa Penyimpanan S3 adalah metrik harian dan dipublikasikan ke CloudWatch sekali per hari. Saat Anda melakukan kueri pada metrik Lensa Penyimpanan S3 CloudWatch, periode kueri harus 1 hari (86400 detik). Setelah metrik S3 Storage Lens harian Anda muncul di dasbor S3 Storage Lens di konsol Amazon S3, diperlukan beberapa jam agar metrik yang sama ini muncul CloudWatch. Saat Anda mengaktifkan opsi CloudWatch penerbitan untuk metrik Lensa Penyimpanan S3 untuk pertama kalinya, dibutuhkan waktu hingga 24 jam untuk dipublikasikan metrik Anda CloudWatch.

Untuk informasi selengkapnya tentang metrik dan dimensi S3 Storage Lens CloudWatch, lihat topik-topik berikut.

Topik

- Metrik
- Dimensi

Metrik

Metrik S3 Storage Lens tersedia sebagai metrik di dalamnya CloudWatch. Metrik S3 Storage Lens dipublikasikan ke`AWS/S3/Storage-Lens` namespace. Namespace ini hanya untuk metrik S3 Storage Lens. Bucket, permintaan, dan metrik replikasi Amazon S3 dipublikasikan ke`AWS/S3` ruang nama.

Metrik S3 Storage Lens dipublikasikan CloudWatch di akun yang memiliki konfigurasi S3 Storage Lens. Setelah Anda mengaktifkan opsi CloudWatch penerbitan dalam metrik dan rekomendasi lanjutan, Anda dapat mengakses metrik tingkat organisasi, akun, dan bucket CloudWatch. Metrik tingkat awalan tidak tersedia di CloudWatch.

Di Lensa Penyimpanan S3, metrik dikumpulkan dan disimpan hanya di Wilayah rumah yang ditentukan. Metrik Lensa Penyimpanan S3 juga dipublikasikan CloudWatch di Wilayah beranda yang Anda tentukan dalam konfigurasi Lensa Penyimpanan S3.

Untuk daftar lengkap metrik Lensa Penyimpanan S3, termasuk daftar metrik yang tersedia CloudWatch, lihatGlosarium metrik Amazon S3 Storage Lens.

> **Note**
>
> Statistik yang valid untuk metrik S3 Storage Lens di dalam CloudWatch adalah Rata-Rata. Untuk informasi selengkapnya tentang statistik CloudWatch, lihat definisi CloudWatch statistik di Panduan CloudWatch Pengguna Amazon.

Granularitas metrik S3 Storage Lens di CloudWatch

S3 Storage Lens menawarkan metrik pada granularitas organisasi, akun, bucket, dan awalan. S3 Storage Lens menerbitkan metrik Lensa Penyimpanan S3 tingkat organisasi, akun, dan ember ke CloudWatch. Metrik Lensa Penyimpanan S3 tingkat awalan tidak tersedia di CloudWatch.

Untuk informasi selengkapnya tentang perincian metrik S3 Storage Lens yang tersedia di dalamnya CloudWatch, lihat daftar berikut:

- Organisasi — Metrik yang dikumpulkan di seluruh akun anggota di organisasi Anda. S3 Storage Lens menerbitkan metrik untuk akun anggota CloudWatch di akun manajemen.
  - Organisasi dan akun — Metrik untuk akun anggota di organisasi Anda.
  - Organisasi dan bucket — Metrik untuk bucket Amazon S3 di akun anggota organisasi Anda.
- Akun (Tingkat non-organisasi) — Metrik digabungkan di seluruh bucket di akun Anda.
- Bucket (Tingkat non-organisasi) — Metrik untuk bucket tertentu. Di dalamnya CloudWatch, S3 Storage Lens menerbitkan metrik ini ke konfigurasi Lensa Penyimpanan S3Akun AWS yang dibuat. S3 Storage Lens menerbitkan metrik ini hanya untuk konfigurasi non-organisasi.

Dimensi

Ketika Lensa Penyimpanan S3 mengirimkan data ke CloudWatch, dimensi dilampirkan ke setiap metrik. Dimensi adalah kategori yang menggambarkan karakteristik metrik. Anda dapat menggunakan dimensi untuk menyaring hasil yang CloudWatch kembali.

Misalnya, semua metrik S3 Storage Lens CloudWatch memiliki`configuration_id` dimensi. Anda dapat menggunakan dimensi ini untuk membedakan antara metrik yang terkait dengan konfigurasi Lensa Penyimpanan S3 tertentu. `organization_id`Mengidentifikasi metrik tingkat organisasi. Untuk informasi selengkapnya tentang dimensi CloudWatch, lihat Dimensi di PanduanCloudWatch Pengguna.

Dimensi yang berbeda tersedia untuk metrik Lensa Penyimpanan S3 tergantung pada perincian metrik. Misalnya, Anda dapat menggunakan`organization_id` dimensi untuk memfilter metrik tingkat organisasi berdasarkanAWS Organizations ID. Namun, Anda tidak dapat menggunakan dimensi ini untuk metrik bucket dan tingkat akun. Untuk informasi selengkapnya, lihat Metrik penyaringan menggunakan dimensi.

Untuk melihat dimensi mana yang tersedia untuk konfigurasi Lensa Penyimpanan S3 Anda, lihat tabel berikut.

Dimensi	Deskripsi	Embe	Akun	Organisasi dan akun	Organisasi dan embe	Organisasi
`configuration_id`	Nama dasbor untuk konfigurasi Lensa Penyimpanan S3 yang dilaporkan dalam metrik	.	.	.	.	.
`metrics_version`	Versi metrik S3 Storage Lens. Versi metrik memiliki nilai tetap`1.0`.	.	.	.	.	.
`organization_id`	AWS OrganizationsID untuk metrik	.	.	.	.	.
`aws_account_number`	Akun AWSYang terkait dengan metrik	.	.	.	.	.
`aws_region`	Wilayah AWSUntuk metrik	.	.	.	.	.
`bucket_name`	Nama bucket S3 yang dilaporkan dalam metrik	.	.	.	.	.
`storage_class`	Kelas penyimpanan untuk bucket yang dilaporkan dalam metrik	.	.	.	.	.
`record_type`	Granularitas metrik: ORGANISASI, AKUN, BUCKET	EMBAKU	AKUN	EMBAKU	AKU	ORGANISASI

Mengaktifkan CloudWatch penerbitan untuk S3 Storage Lens

Anda dapat mempublikasikan metrik S3 Storage Lens CloudWatch ke Amazon untuk membuat tampilan terpadu kesehatan operasional Anda di CloudWatchdasbor. Anda juga dapat menggunakan CloudWatch fitur, seperti alarm dan tindakan yang dipicu, matematika metrik, dan deteksi anomali, untuk memantau dan mengambil tindakan pada metrik Lensa Penyimpanan S3. Selain itu, operasi CloudWatch API memungkinkan aplikasi, termasuk penyedia pihak ketiga, untuk mengakses metrik Lensa Penyimpanan S3 Anda. Untuk informasi selengkapnya tentang CloudWatch fitur, lihat Panduan CloudWatch Pengguna Amazon.

Metrik S3 Storage Lens dipublikasikan CloudWatch di akun yang memiliki konfigurasi S3 Storage Lens. Setelah Anda mengaktifkan opsi CloudWatch penerbitan dalam metrik dan rekomendasi lanjutan, Anda dapat mengakses metrik tingkat organisasi, akun, dan bucket CloudWatch. Metrik tingkat awalan tidak tersedia di CloudWatch.

Anda dapat mengaktifkan CloudWatch dukungan untuk konfigurasi dasbor baru atau yang sudah ada dengan menggunakan konsol S3, Amazon S3 REST APIAWS CLI, danAWS SDK. Opsi CloudWatch penerbitan tersedia untuk dasbor yang ditingkatkan ke metrik dan rekomendasi lanjutan S3 Storage Lens. Untuk harga metrik dan rekomendasi tingkat lanjut S3 Storage Lens, lihat harga Amazon S3. Tidak ada biaya penerbitan CloudWatch metrik tambahan yang berlaku; namun, CloudWatch biaya lain, seperti dasbor, alarm, dan panggilan API, berlaku.

Untuk mengaktifkan opsi CloudWatch penerbitan metrik Lensa Penyimpanan S3, lihat topik berikut.

> **Note**
>
> Metrik Lensa Penyimpanan S3 adalah metrik harian dan dipublikasikan ke CloudWatch sekali per hari. Saat Anda melakukan kueri pada metrik Lensa Penyimpanan S3 CloudWatch, periode kueri harus 1 hari (86400 detik). Setelah metrik S3 Storage Lens harian Anda muncul di dasbor S3 Storage Lens di konsol Amazon S3, diperlukan beberapa jam agar metrik yang sama ini muncul CloudWatch. Saat Anda mengaktifkan opsi CloudWatch penerbitan untuk metrik Lensa Penyimpanan S3 untuk pertama kalinya, dibutuhkan waktu hingga 24 jam untuk dipublikasikan metrik Anda CloudWatch.
> Saat ini, metrik Lensa Penyimpanan S3 tidak dapat dikonsumsi melalui CloudWatch aliran.

Menggunakan konsol S3

Saat memperbarui dasbor S3 Storage Lens, Anda tidak dapat mengubah nama dasbor atau Wilayah asal. Anda juga tidak dapat mengubah cakupan dasbor default, yang mencakup seluruh penyimpanan akun Anda.

Untuk memperbarui dasbor S3 Storage Lens untuk mengaktifkan CloudWatch penerbitan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih S3 Storage Lens, Dasbor.
3. Pilih dasbor yang ingin Anda edit, lalu pilih Edit.
4. Di bawah Pemilihan metrik, pilih Metrik dan rekomendasi lanjutan.

   Metrik dan rekomendasi tingkat lanjut tersedia dengan biaya tambahan. Metrik dan rekomendasi tingkat lanjut mencakup periode 15 bulan untuk kueri data, metrik penggunaan yang digabungkan di tingkat prefiks, metrik aktivitas yang digabungkan berdasarkan bucket, opsi CloudWatch penerbitan, dan rekomendasi kontekstual yang membantu Anda mengoptimalkan biaya penyimpanan, dan menerapkan praktik terbaik perlindungan data. Untuk informasi selengkapnya, lihat Harga Amazon S3.
5. Di bawah Pilih Fitur metrik dan rekomendasi lanjutan, pilih CloudWatchpenerbitan.

   > **Important**
   >
   > Jika konfigurasi Anda mengaktifkan agregasi awalan untuk metrik penggunaan, metrik tingkat awalan tidak akan dipublikasikan CloudWatch. Hanya metrik bucket, akun, dan metrik S3 tingkat organisasi tingkat organisasi yang dipublikasikan CloudWatch.
6. Pilih Save changes (Simpan perubahan).

Untuk membuat dasbor S3 Storage Lens baru yang memungkinkan CloudWatch dukungan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Storage Lens, Dasbor.
3. Pilih Create dashboard (Buat dasbor).
4. Di bawah Umum, tentukan opsi konfigurasi berikut:

a. Untuk nama Dasbor, masukkan nama dasbor Anda.

   Nama dasbor harus kurang dari 65 karakter dan tidak boleh berisi karakter khusus atau spasi. Anda tidak dapat mengubah nama dasbor setelah Anda membuat dasbor.

b. Pilih Wilayah Asal untuk dasbor Anda.

   Metrik untuk semua Wilayah yang disertakan dalam lingkup dasbor ini disimpan secara terpusat di Wilayah asal yang ditunjuk. Di dalamnya CloudWatch, metrik Lensa Penyimpanan S3 juga tersedia di Wilayah asal. Anda tidak dapat mengubah Wilayah asal setelah Anda membuat dasbor.

5. (Opsional) Untuk menambahkan tag, pilih Tambahkan tag dan masukkan tag Key dan Value.

> **Note**
>
> Anda dapat menambahkan hingga 50 tag ke konfigurasi dasbor.

6. Tentukan ruang lingkup untuk konfigurasi Anda:

a. Jika Anda membuat konfigurasi tingkat organisasi, pilih akun yang akan disertakan dalam konfigurasi: Sertakan semua akun dalam konfigurasi Anda atau Batasi cakupan ke akun Anda yang masuk.

> **Note**
>
> Bila Anda membuat konfigurasi tingkat organisasi yang mencakup semua akun, Anda dapat menyertakan atau mengecualikan hanya Wilayah, bukan bucket.

b. Pilih Wilayah dan bucket yang Anda inginkan untuk disertakan dalam konfigurasi dasbor dengan melakukan hal berikut:

   - Untuk menyertakan semua Wilayah, pilih Sertakan Wilayah dan bucket.
   - Untuk menyertakan Wilayah tertentu, hapus Sertakan semua Wilayah. Di bawah Pilih Wilayah untuk disertakan, pilih Wilayah yang Anda inginkan untuk disertakan di dasbor.
   - Untuk menyertakan ember tertentu, kosongkan Sertakan semua ember. Di bawah Pilih bucket untuk disertakan, pilih bucket yang Anda inginkan untuk disertakan di dasbor.

> **Note**
>
> Anda dapat memilih hingga 50 bucket.

7. Untuk pemilihan Metrik, pilih Metrik dan rekomendasi lanjutan.

   Untuk informasi selengkapnya tentang harga metrik dan rekomendasi tingkat lanjut, lihat harga Amazon S3.

8. Di bawah Fitur metrik dan rekomendasi lanjutan, pilih opsi yang ingin Anda aktifkan:

   - Metrik tingkat lanjut
   - CloudWatch penerbitan

   > **Important**
   >
   > Jika Anda mengaktifkan agregasi awalan untuk konfigurasi Lensa Penyimpanan S3, metrik tingkat awalan tidak akan dipublikasikan CloudWatch. Hanya metrik bucket, akun, dan metrik S3 tingkat organisasi tingkat organisasi yang dipublikasikan CloudWatch.

   - Agregasi awalan

   > **Note**
   >
   > Untuk informasi selengkapnya tentang fitur metrik dan rekomendasi lanjutan, lihatPilihan metrik.

9. Jika Anda mengaktifkan Metrik lanjutan, pilih kategori Metrik lanjutan yang ingin ditampilkan di dasbor Lensa Penyimpanan S3 Anda:

   - Metrik aktivitas
   - Metrik kode status terperinci
   - Metrik pengoptimalan biaya tingkat lanjut
   - Metrik perlindungan data tingkat lanjut

Untuk informasi selengkapnya tentang kategori metrik, lihatKategori metrik. Untuk daftar lengkap metrik, lihatGlosarium metrik Amazon S3 Storage Lens.

10. (Opsional) Konfigurasikan ekspor metrik Anda.

    Untuk informasi selengkapnya tentang cara mengonfigurasi ekspor metrik, lihat langkahMembuat dasbor Lensa Penyimpanan Amazon S3.

11. Pilih Create dashboard (Buat dasbor).

Menggunakan AWS CLI

AWS CLIContoh berikut memungkinkan opsi CloudWatch penerbitan dengan menggunakan konfigurasi metrik dan rekomendasi tingkat lanjut S3 Storage Lens. Untuk menggunakan contoh ini, ganti*`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control put-storage-lens-configuration --account-id=555555555555 --config-
id=your-configuration-id --region=us-east-1 --storage-lens-configuration=file://./
config.json

config.json
{
  "Id": "SampleS3StorageLensConfiguration", //Use this property to identify your S3
 Storage Lens configuration.
  "AwsOrg": { //Use this property when enabling S3 Storage Lens for AWS Organizations.
    "Arn": "arn:aws:organizations::123456789012:organization/o-abcdefgh"
  },
  "AccountLevel": {
    "ActivityMetrics": {
      "IsEnabled":true
    },
    "AdvancedCostOptimizationMetrics": {
      "IsEnabled":true
    },
    "AdvancedDataProtectionMetrics": {
      "IsEnabled":true
    },
    "DetailedStatusCodesMetrics": {
      "IsEnabled":true
    },
    "BucketLevel": {
      "ActivityMetrics": {
```

```
        "IsEnabled":true //Mark this as false if you want only free metrics.
      },
      "ActivityMetrics": {
        "IsEnabled":true //Mark this as false if you want only free metrics.
      },
      "AdvancedCostOptimizationMetrics": {
        "IsEnabled":true //Mark this as false if you want only free metrics.
      },
      "DetailedStatusCodesMetrics": {
        "IsEnabled":true //Mark this as false if you want only free metrics.
      },
      "PrefixLevel":{
        "StorageMetrics":{
          "IsEnabled":true, //Mark this as false if you want only free metrics.
          "SelectionCriteria":{
            "MaxDepth":5,
            "MinStorageBytesPercentage":1.25,
            "Delimiter":"/"
          }
        }
      }
    }
  },
  "Exclude": { //Replace with "Include" if you prefer to include Regions.
    "Regions": [
      "eu-west-1"
    ],
    "Buckets": [ //This attribute is not supported for AWS Organizations-level
 configurations.
      "arn:aws:s3:::source_bucket1"
    ]
  },
  "IsEnabled": true, //Whether the configuration is enabled
  "DataExport": { //Details about the metrics export
    "S3BucketDestination": {
      "OutputSchemaVersion": "V_1",
      "Format": "CSV", //You can add "Parquet" if you prefer.
      "AccountId": "111122223333",
      "Arn": "arn:aws:s3:::destination-bucket-name", // The destination bucket for your
 metrics export must be in the same Region as your S3 Storage Lens configuration.
      "Prefix": "prefix-for-your-export-destination",
      "Encryption": {
        "SSES3": {}
      }
```

```
    },
    "CloudWatchMetrics": {
      "IsEnabled": true //Mark this as false if you want to export only free metrics.
    }
  }
}
```

MenggunakanAWS SDK for Java

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.AccountLevel;
import com.amazonaws.services.s3control.model.ActivityMetrics;
import com.amazonaws.services.s3control.model.BucketLevel;
import com.amazonaws.services.s3control.model.CloudWatchMetrics;
import com.amazonaws.services.s3control.model.Format;
import com.amazonaws.services.s3control.model.Include;
import com.amazonaws.services.s3control.model.OutputSchemaVersion;
import com.amazonaws.services.s3control.model.PrefixLevel;
import com.amazonaws.services.s3control.model.PrefixLevelStorageMetrics;
import com.amazonaws.services.s3control.model.PutStorageLensConfigurationRequest;
import com.amazonaws.services.s3control.model.S3BucketDestination;
import com.amazonaws.services.s3control.model.SSES3;
import com.amazonaws.services.s3control.model.SelectionCriteria;
import com.amazonaws.services.s3control.model.StorageLensAwsOrg;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;
import com.amazonaws.services.s3control.model.StorageLensDataExport;
import com.amazonaws.services.s3control.model.StorageLensDataExportEncryption;
import com.amazonaws.services.s3control.model.StorageLensTag;

import java.util.Arrays;
import java.util.List;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class CreateAndUpdateDashboard {

    public static void main(String[] args) {
```

```
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";
        String exportAccountId = "Destination Account ID";
        String exportBucketArn = "arn:aws:s3:::destBucketName"; // The destination
 bucket for your metrics export must be in the same Region as your S3 Storage Lens
 configuration.
        String awsOrgARN = "arn:aws:organizations::123456789012:organization/o-
abcdefgh";
        Format exportFormat = Format.CSV;

        try {
            SelectionCriteria selectionCriteria = new SelectionCriteria()
                    .withDelimiter("/")
                    .withMaxDepth(5)
                    .withMinStorageBytesPercentage(10.0);
            PrefixLevelStorageMetrics prefixStorageMetrics = new
 PrefixLevelStorageMetrics()
                    .withIsEnabled(true)
                    .withSelectionCriteria(selectionCriteria);
            BucketLevel bucketLevel = new BucketLevel()
                    .withActivityMetrics(new ActivityMetrics().withIsEnabled(true))
                    .withAdvancedCostOptimizationMetrics(new
 AdvancedCostOptimizationMetrics().withIsEnabled(true))
                    .withAdvancedDataProtectionMetrics(new
 AdvancedDataProtectionMetrics().withIsEnabled(true))
                    .withDetailedStatusCodesMetrics(new
 DetailedStatusCodesMetrics().withIsEnabled(true))
                    .withPrefixLevel(new
 PrefixLevel().withStorageMetrics(prefixStorageMetrics));
            AccountLevel accountLevel = new AccountLevel()
                    .withActivityMetrics(new ActivityMetrics().withIsEnabled(true))
                    .withAdvancedCostOptimizationMetrics(new
 AdvancedCostOptimizationMetrics().withIsEnabled(true))
                    .withAdvancedDataProtectionMetrics(new
 AdvancedDataProtectionMetrics().withIsEnabled(true))
                    .withDetailedStatusCodesMetrics(new
 DetailedStatusCodesMetrics().withIsEnabled(true))
                    .withBucketLevel(bucketLevel);

            Include include = new Include()
                    .withBuckets(Arrays.asList("arn:aws:s3:::bucketName"))
                    .withRegions(Arrays.asList("us-west-2"));
```

```
            StorageLensDataExportEncryption exportEncryption = new
StorageLensDataExportEncryption()
                    .withSSES3(new SSES3());
            S3BucketDestination s3BucketDestination = new S3BucketDestination()
                    .withAccountId(exportAccountId)
                    .withArn(exportBucketArn)
                    .withEncryption(exportEncryption)
                    .withFormat(exportFormat)
                    .withOutputSchemaVersion(OutputSchemaVersion.V_1)
                    .withPrefix("Prefix");
            CloudWatchMetrics cloudWatchMetrics = new CloudWatchMetrics()
                    .withIsEnabled(true);
            StorageLensDataExport dataExport = new StorageLensDataExport()
                    .withCloudWatchMetrics(cloudWatchMetrics)
                    .withS3BucketDestination(s3BucketDestination);

            StorageLensAwsOrg awsOrg = new StorageLensAwsOrg()
                    .withArn(awsOrgARN);

            StorageLensConfiguration configuration = new StorageLensConfiguration()
                    .withId(configurationId)
                    .withAccountLevel(accountLevel)
                    .withInclude(include)
                    .withDataExport(dataExport)
                    .withAwsOrg(awsOrg)
                    .withIsEnabled(true);

            List<StorageLensTag> tags = Arrays.asList(
                    new StorageLensTag().withKey("key-1").withValue("value-1"),
                    new StorageLensTag().withKey("key-2").withValue("value-2")
            );

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.putStorageLensConfiguration(new
PutStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withStorageLensConfiguration(configuration)
                    .withTags(tags)
            );
```

```
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Penggunaan REST API

Untuk mengaktifkan opsi CloudWatch penerbitan dengan menggunakan Amazon S3 REST API, Anda dapat menggunakannya PutStorageLensConfiguration.

Langkah selanjutnya

Setelah mengaktifkan opsi CloudWatch penerbitan, Anda dapat mengakses metrik Lensa Penyimpanan S3 Anda CloudWatch. Anda juga dapat memanfaatkan CloudWatch fitur untuk memantau dan menganalisis data Lensa Penyimpanan S3 Anda CloudWatch. Untuk informasi lain, lihat topik berikut:

- Metrik dan dimensi S3 Storage Lens
- Bekerja dengan metrik S3 Storage Lens di CloudWatch

Bekerja dengan metrik S3 Storage Lens di CloudWatch

Anda dapat mempublikasikan metrik S3 Storage Lens CloudWatch ke Amazon untuk membuat tampilan terpadu kesehatan operasional Anda di CloudWatchdasbor. Anda juga dapat menggunakan CloudWatch fitur, seperti alarm dan tindakan yang dipicu, matematika metrik, dan deteksi anomali, untuk memantau dan mengambil tindakan pada metrik Lensa Penyimpanan S3. Selain itu, operasi CloudWatch API memungkinkan aplikasi, termasuk penyedia pihak ketiga, untuk mengakses metrik Lensa Penyimpanan S3 Anda. Untuk informasi selengkapnya tentang CloudWatch fitur, lihat Panduan CloudWatch Pengguna Amazon.

Anda dapat mengaktifkan opsi CloudWatch penerbitan untuk konfigurasi dasbor baru atau yang sudah ada menggunakan konsol Amazon S3, API REST Amazon S3,AWS CLI, danAWS SDKs. Opsi CloudWatch penerbitan tersedia untuk dasbor yang ditingkatkan ke metrik dan rekomendasi

lanjutan S3 Storage Lens. Untuk harga metrik dan rekomendasi tingkat lanjut S3 Storage Lens, lihat harga Amazon S3. Tidak ada biaya penerbitan CloudWatch metrik tambahan yang berlaku; namun, CloudWatch biaya lain, seperti dasbor, alarm, dan panggilan API, berlaku. Untuk informasi selengkapnya, lihat CloudWatch harga Amazon.

Metrik S3 Storage Lens dipublikasikan CloudWatch di akun yang memiliki konfigurasi S3 Storage Lens. Setelah Anda mengaktifkan opsi CloudWatch penerbitan dalam metrik dan rekomendasi lanjutan, Anda dapat mengakses metrik tingkat organisasi, akun, dan bucket CloudWatch. Metrik tingkat awalan tidak tersedia di CloudWatch.

> **Note**
>
> Metrik Lensa Penyimpanan S3 adalah metrik harian dan dipublikasikan ke CloudWatch sekali per hari. Saat Anda melakukan kueri pada metrik Lensa Penyimpanan S3 CloudWatch, periode kueri harus 1 hari (86400 detik). Setelah metrik S3 Storage Lens harian Anda muncul di dasbor S3 Storage Lens di konsol Amazon S3, diperlukan beberapa jam agar metrik yang sama ini muncul CloudWatch. Saat Anda mengaktifkan opsi CloudWatch penerbitan untuk metrik Lensa Penyimpanan S3 untuk pertama kalinya, dibutuhkan waktu hingga 24 jam untuk dipublikasikan metrik Anda CloudWatch.
> Saat ini, metrik Lensa Penyimpanan S3 tidak dapat dikonsumsi melalui CloudWatch aliran.

Untuk informasi selengkapnya tentang bekerja dengan metrik S3 Storage Lens CloudWatch, lihat topik-topik berikut.

Topik

- Bekerja dengan CloudWatch dasbor
- Mengatur alarm, memicu tindakan, dan menggunakan deteksi anomali
- Metrik penyaringan menggunakan dimensi
- Menghitung metrik baru dengan matematika metrik
- Menggunakan ekspresi pencarian di grafik

Bekerja dengan CloudWatch dasbor

Anda dapat menggunakan CloudWatch dasbor untuk memantau metrik Lensa Penyimpanan S3 bersama metrik aplikasi lainnya dan membuat tampilan terpadu tentang kesehatan operasional

Anda. Dasbor adalah halaman beranda yang dapat disesuaikan di konsol yang dapat disesuaikan di CloudWatch konsol yang dapat disesuaikan di konsol yang dapat disesuaikan di konsol yang dapat disesuaikan di konsol yang dapat disesuaikan di konsol yang dapat disesuaikan di konsol yang dapat disesuaikan.

CloudWatch memiliki kontrol izin luas yang tidak mendukung pembatasan akses ke serangkaian metrik atau dimensi tertentu. Pengguna di akun atau organisasi Anda yang memiliki akses CloudWatch akan memiliki akses ke metrik untuk semua konfigurasi Lensa Penyimpanan S3 di mana opsi CloudWatch dukungan diaktifkan. Anda tidak dapat mengelola izin untuk dasbor tertentu seperti yang Anda bisa di S3 Storage Lens. Untuk informasi selengkapnya tentang CloudWatch izin, lihat Mengelola izin akses ke CloudWatch sumber daya Anda di Panduan CloudWatch Pengguna Amazon.

Untuk informasi selengkapnya tentang penggunaan CloudWatch dasbor dan mengonfigurasi izin, lihat Menggunakan CloudWatch dasbor Amazon dan CloudWatch Dasbor Berbagi di Panduan CloudWatch Pengguna Amazon.

Mengatur alarm, memicu tindakan, dan menggunakan deteksi anomali

Anda dapat mengonfigurasi CloudWatch alarm yang menonton metrik Lensa Penyimpanan S3 CloudWatch dan mengambil tindakan saat ambang batas dilanggar. Misalnya, Anda dapat mengonfigurasi alarm yang mengirimkan notifikasi Amazon SNS saat metrik Uncomplete Multipart Upload Bytes melebihi 1 GB selama tiga hari berturut-turut.

Anda juga dapat mengaktifkan deteksi anomali untuk terus menerus menganalisis metrik S3 Storage Lens, menentukan garis dasar normal, serta anomali permukaan. Anda dapat membuat alarm deteksi anomali berdasarkan nilai yang diharapkan dari metrik. Misalnya, Anda dapat memantau anomali untuk metrik Object Lock Enabled Bytes untuk mendeteksi penghapusan pengaturan Object Lock yang tidak sah.

Untuk informasi dan contoh selengkapnya, lihat Menggunakan CloudWatch alarm Amazon dan Membuat alarm dari metrik pada grafik di Panduan CloudWatch Pengguna Amazon.

Metrik penyaringan menggunakan dimensi

Anda dapat menggunakan dimensi untuk menyaring metrik S3 Storage Lens di CloudWatch konsol. Misalnya, Anda dapat memfilter berdasarkan`configuration_id`,,`aws_account_number`,`aws_regionbucket_name`, dan banyak lagi.

S3 Storage Lens mendukung beberapa konfigurasi dasbor per akun. Ini berarti bahwa konfigurasi yang berbeda dapat mencakup bucket yang sama. Saat metrik ini dipublikasikan CloudWatch, bucket akan memiliki metrik duplikat di dalamnya CloudWatch. Untuk melihat metrik hanya untuk konfigurasi Lensa Penyimpanan S3 tertentu CloudWatch, Anda dapat menggunakan`configuration_id` dimensi. Saat Anda memfilter menurut`configuration_id`, Anda hanya melihat metrik yang terkait dengan konfigurasi yang Anda identifikasi.

Untuk informasi selengkapnya tentang pemfilteran menurut ID konfigurasi, lihat Mencari metrik yang tersedia di Panduan CloudWatch Pengguna Amazon.

Menghitung metrik baru dengan matematika metrik

Anda dapat menggunakan matematika metrik untuk membuat kueri beberapa metrik S3 Storage Lens dan menggunakan ekspresi matematika untuk membuat deret waktu baru berdasarkan metrik ini. Misalnya, Anda dapat membuat metrik baru untuk objek yang tidak terenkripsi dengan mengurangi Objek Terenkripsi dari Objek Hitungan. Anda juga dapat membuat metrik untuk mendapatkan ukuran objek rata-rata dengan membagi`StorageBytes` dengan`ObjectCount` atau jumlah byte yang diakses pada satu hari`BytesDownloaded` dengan membaginya`StorageBytes`.

Untuk informasi selengkapnya, lihat Menggunakan matematika metrik di Panduan CloudWatch Pengguna Amazon.

Menggunakan ekspresi pencarian di grafik

Dengan metrik Lensa Penyimpanan S3, Anda dapat membuat ekspresi pencarian. Misalnya, Anda dapat membuat ekspresi pencarian untuk semua metrik yang diberi nama IncompleteMultipartUploadStorageBytesdan`SUM` menambah ekspresi. Dengan ekspresi pencarian ini, Anda dapat melihat total byte unggahan multibagian yang tidak lengkap di semua dimensi penyimpanan Anda dalam satu metrik.

Contoh ini menunjukkan sintaks yang akan Anda gunakan untuk membuat ekspresi pencarian untuk semua metrik bernama IncompleteMultipartUploadStorageBytes.

```
SUM(SEARCH('{AWS/S3/Storage-
Lens,aws_account_number,aws_region,configuration_id,metrics_version,record_type,storage_class}
 MetricName="IncompleteMultipartUploadStorageBytes"', 'Average',86400))
```

Untuk informasi selengkapnya tentang sintaks ini, lihat sintaks ekspresiCloudWatch pencarian di Panduan CloudWatch Pengguna Amazon. Untuk membuat CloudWatch grafik dengan ekspresi

pencarian, lihat Membuat CloudWatch grafik dengan ekspresi pencarian di Panduan CloudWatch Pengguna Amazon.

Amazon S3

Anda dapat menggunakan dasbor Amazon S3, untuk memvisualisasikan wawasan dan tren, menandai outer, dan menerima rekomendasi. Metrik Lensa Penyimpanan S3 disusun ke dalam kategori yang selaras dengan kasus penggunaan utama. Anda dapat menggunakan metrik ini untuk melakukan hal berikut:

- Identifikasi peluang pengoptimalan biaya
- Menerapkan praktik terbaik perlindungan data
- Menerapkan praktik terbaik manajemen akses
- Meningkatkan kinerja beban kerja aplikasi

Misalnya, dengan metrik pengoptimalan biaya, Anda dapat mengidentifikasi peluang untuk mengurangi biaya penyimpanan Amazon S3. Anda dapat mengidentifikasi bucket dengan unggahan multibagian yang berusia lebih dari 7 hari atau bucket yang mengumpulkan versi noncurrent.

Demikian pula, Anda dapat menggunakan metrik perlindungan data untuk mengidentifikasi bucket yang tidak mengikuti praktik terbaik perlindungan data dalam organisasi Anda. Misalnya, Anda dapat mengidentifikasi bucket yang tidak menggunakanAWS Key Management Service kunci (SSE-KMS) untuk enkripsi default atau tidak mengaktifkan Versi S3.

Dengan metrik manajemen akses Lensa Penyimpanan S3, Anda dapat mengidentifikasi setelan bucket untuk Kepemilikan Objek S3 sehingga Anda dapat memigrasikan izin daftar kontrol akses (ACL) ke kebijakan bucket dan menonaktifkan ACL.

Jika Anda mengaktifkan metrik lanjutan Lensa Penyimpanan S3, Anda dapat menggunakan metrik kode status terperinci untuk mendapatkan jumlah permintaan yang berhasil atau gagal yang dapat Anda gunakan untuk memecahkan masalah akses atau kinerja.

Dengan metrik lanjutan, Anda juga dapat mengakses metrik pengoptimalan biaya dan perlindungan data tambahan yang dapat Anda gunakan untuk mengidentifikasi peluang guna mengurangi biaya penyimpanan S3 Anda secara keseluruhan dan lebih selaras dengan praktik terbaik untuk melindungi data Anda. Misalnya, metrik pengoptimalan biaya tingkat lanjut menyertakan jumlah aturan siklus hidup yang dapat Anda gunakan untuk mengidentifikasi bucket yang tidak memiliki aturan siklus

hidup untuk mengakhiri upload multibagian yang tidak lengkap yang berusia lebih dari 7 hari. Metrik perlindungan data tingkat lanjut mencakup jumlah aturan replikasi.

Untuk informasi lebih lanjut tentang kategori metrik, lihatKategori metrik. Untuk daftar lengkap metrik Lensa Penyimpanan S3, lihatGlosarium metrik Amazon S3 Storage Lens.

Topik

- Menggunakan Amazon S3 Storage Lens untuk mengoptimalkan biaya penyimpanan Anda
- Menggunakan Lensa Penyimpanan S3 untuk melindungi data Anda
- Menggunakan S3 Storage Lens untuk mengaudit pengaturan Object Ownership
- Menggunakan metrik S3 Storage Lens untuk meningkatkan kinerja

Menggunakan Amazon S3 Storage Lens untuk mengoptimalkan biaya penyimpanan Anda

Anda dapat menggunakan metrik pengoptimalan biaya Lensa Penyimpanan S3 untuk mengurangi biaya penyimpanan S3 Anda secara keseluruhan. Metrik pengoptimalan biaya dapat membantu Anda mengonfirmasi bahwa Anda telah mengonfigurasi biaya Amazon S3 secara efektif dan sesuai dengan praktik terbaik. Misalnya, Anda dapat mengidentifikasi peluang pengoptimalan biaya berikut:

- Bucket dengan unggahan multibagian yang tidak lengkap lebih dari 7 hari
- Ember yang mengumpulkan banyak versi noncurrent
- Bucket yang tidak memiliki aturan siklus hidup untuk membatalkan unggahan multibagian yang tidak lengkap
- Bucket yang tidak memiliki aturan siklus hidup untuk berakhir objek versi noncurrent
- Bucket yang tidak memiliki aturan siklus hidup untuk mentransisikan objek ke kelas penyimpanan yang berbeda

Anda kemudian dapat menggunakan data ini untuk menambahkan aturan siklus hidup tambahan ke bucket Anda.

Contoh berikut menunjukkan bagaimana Anda dapat menggunakan metrik pengoptimalan biaya di dasbor S3 Storage Lens untuk mengoptimalkan biaya penyimpanan Anda.

Topik

- Identifikasi bucket S3 terbesar Anda

- Temukan ember Amazon S3 yang dingin
- Temukan unggahan multibagian yang tidak lengkap
- Kurangi jumlah versi noncurrent yang dipertahankan
- Identifikasi bucket yang tidak memiliki aturan siklus hidup dan tinjau jumlah aturan siklus hidup

Identifikasi bucket S3 terbesar Anda

Anda membayar untuk menyimpan objek dalam ember S3. Tarif yang dikenakan biaya tergantung pada ukuran objek Anda, berapa lama Anda menyimpan objek, dan kelas penyimpanannya. Dengan S3 Storage Lens, Anda mendapatkan tampilan terpusat dari semua bucket di akun Anda. Untuk melihat semua bucket di semua akun organisasi Anda, Anda dapat mengonfigurasiAWS Organizations-tingkat S3 Storage Lens dashboard. Dari tampilan dasbor ini, Anda dapat mengidentifikasi bucket terbesar Anda.

Langkah 1: Identifikasi ember terbesar Anda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihLensa Penyimpanan,Dasbor.
3. Di daftar Dasbor, pilih dasbor yang ingin Anda lihat.

   Saat dasbor terbuka, Anda dapat melihat tanggal terakhir S3 Storage Lens telah mengumpulkan metrik. Dasbor Anda selalu dimuat ke tanggal terbaru yang memiliki metrik yang tersedia.
4. Untuk melihat peringkat bucket terbesar Anda olehTotal penyimpananmetrik untuk rentang tanggal yang dipilih, gulir ke bawah keIkhtisar N teratas untukkencanbagian.

   Anda dapat beralih urutan untuk menampilkan ember terkecil. Anda juga dapat menyesuaikanMetrikpilihan untuk menentukan peringkat bucket Anda dengan salah satu metrik yang tersedia. YangIkhtisar N teratas untukkencanbagian juga menunjukkan persentase perubahan dari hari sebelumnya atau minggu dan spark-line untuk memvisualisasikan tren. Tren ini adalah tren 14 hari untuk metrik gratis dan tren 30 hari untuk metrik dan rekomendasi lanjutan.

   > **Note**
   >
   > Dengan metrik dan rekomendasi lanjutan Lensa Penyimpanan S3, metrik tersedia untuk kueri selama 15 bulan. Untuk informasi selengkapnya, lihat Pilihan metrik.

5. Untuk wawasan lebih rinci tentang bucket Anda, gulir ke atas ke bagian atas halaman, lalu pilihEmbertab.

   PadaEmbertab, Anda dapat melihat rincian seperti tingkat pertumbuhan baru-baru ini, ukuran objek rata-rata, awalan terbesar, dan jumlah objek.

Langkah 2: Arahkan ke ember Anda dan selidiki

Setelah mengidentifikasi bucket S3 terbesar, Anda dapat menavigasi ke setiap bucket dalam konsol S3 untuk melihat objek di bucket, memahami beban kerja terkait, dan mengidentifikasi pemilik internalnya. Anda dapat menghubungi pemilik bucket untuk mengetahui apakah pertumbuhan diharapkan atau apakah pertumbuhan membutuhkan pemantauan dan kontrol lebih lanjut.

Temukan ember Amazon S3 yang dingin

Jika Anda memilikiMetrik lanjutan Lensa Penyimpanan S3diaktifkan, Anda dapat menggunakanmetrik aktivitasuntuk memahami seberapa dingin ember S3 Anda. Ember “dingin” adalah ember yang penyimpanannya tidak lagi diakses (atau sangat jarang diakses). Kurangnya aktivitas ini biasanya menunjukkan bahwa objek bucket tidak sering diakses.

Metrik aktivitas, sepertiGET PermintaandanUnduh Bytes, tunjukkan seberapa sering ember Anda diakses setiap hari. Untuk memahami konsistensi pola akses dan untuk melihat bucket yang tidak lagi diakses sama sekali, Anda dapat membuat tren data ini selama beberapa bulan. YangTingkat pengambilanmetrik, yang dihitung sebagaiUnduh bytes/Total storage, menunjukkan proporsi penyimpanan dalam ember yang diakses setiap hari.

> **Note**
>
> Download byte diduplikasi dalam kasus di mana objek yang sama diunduh beberapa kali di siang hari.

Prasyarat

Untuk melihat metrik aktivitas di dasbor Lensa Penyimpanan S3, Anda harus mengaktifkan Lensa Penyimpanan S3Metrik dan rekomendasi tingkat lanjutdan kemudian pilihMetrik aktivitas. Untuk informasi selengkapnya, lihat Membuat dan memperbarui dasbor Lensa Penyimpanan Amazon S3.

Langkah 1: Identifikasi ember aktif

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihLensa Penyimpanan,Dasbor.
3. Di daftar Dasbor, pilih dasbor yang ingin Anda lihat.
4. PilihEmbertab, dan kemudian gulir ke bawah keAnalisis gelembung oleh ember untukkencanbagian.

   Di dalamAnalisis gelembung oleh ember untukkencanbagian, Anda dapat memplot bucket Anda pada beberapa dimensi dengan menggunakan tiga metrik untuk mewakiliSumbu X,Sumbu Y, danUkurangelembung.
5. Untuk menemukan ember yang sudah dingin, untukSumbu X,Sumbu Y, danUkuran, pilihTotal penyimpanan,% tingkat pengambilan, danUkuran objek rata-ratametrik.
6. Di dalamAnalisis gelembung oleh ember untukkencanbagian, cari bucket dengan tingkat pengambilan nol (atau mendekati nol) dan ukuran penyimpanan relatif yang lebih besar, dan pilih gelembung yang mewakili bucket.

   Sebuah kotak akan muncul dengan pilihan untuk wawasan yang lebih terperinci. Lakukan salah satu dari berikut:

   a. Untuk memperbaruiEmbertab untuk menampilkan metrik hanya untuk bucket yang dipilih, pilihMenelusuri, dan kemudian pilihTerapkan.
   b. Untuk menggabungkan data level bucket Anda ke akun,Wilayah AWS, kelas penyimpanan, atau ember, pilihMenganalisis olehdan kemudian membuat pilihan untukDimensi. Misalnya, untuk agregat berdasarkan kelas penyimpanan, pilihKelas penyimpananuntukDimensi.

   Untuk menemukan ember yang sudah dingin, lakukan analisis gelembung menggunakanTotal penyimpanan,% tingkat pengambilan, danUkuran objek rata-ratametrik. Cari bucket apa pun dengan tingkat pengambilan nol (atau mendekati nol) dan ukuran penyimpanan relatif yang lebih besar.

   YangEmbertab pembaruan dasbor Anda untuk menampilkan data agregasi atau filter yang Anda pilih. Jika Anda digabungkan berdasarkan kelas penyimpanan atau dimensi lain, tab baru itu terbuka di dasbor Anda (misalnya,Kelas penyimpanantab).

Langkah 2: Selidiki ember dingin

Dari sini, Anda dapat mengidentifikasi pemilik ember dingin di akun atau organisasi Anda dan mencari tahu apakah penyimpanan itu masih diperlukan. Anda kemudian dapat mengoptimalkan biaya dengan mengkonfigurasikonfigurasi kedaluwarsa siklus hidupuntuk ember ini atau pengarsipan data di salah satuAmazon S3 Glacierkelas penyimpanan.

Untuk menghindari masalah ember dingin ke depan, Anda bisasecara otomatis mentransisikan data Anda dengan menggunakan konfigurasi Siklus Hidup S3untuk ember Anda, atau Anda dapat mengaktifkanpengarsipan otomatis dengan S3 Intelligent-Tiering.

Anda juga dapat menggunakan langkah 1 untuk mengidentifikasi ember panas. Kemudian, Anda dapat memastikan bahwa ember ini menggunakan yang benarKelas penyimpanan S3untuk memastikan bahwa mereka melayani permintaan mereka paling efektif dalam hal kinerja dan biaya.

Temukan unggahan multibagian yang tidak lengkap

Anda dapat menggunakan unggahan multipart untuk mengunggah objek yang sangat besar (hingga 5 TB) sebagai sekumpulan komponen untuk meningkatkan throughput dan pemulihan yang lebih cepat dari masalah jaringan. Dalam kasus di mana proses upload multipart tidak selesai, bagian yang tidak lengkap tetap berada dalam bucket (dalam keadaan tidak dapat digunakan). Suku cadang yang tidak lengkap ini dikenakan biaya penyimpanan sampai proses pengunggahan selesai, atau sampai bagian yang tidak lengkap dilepas. Untuk informasi selengkapnya, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.

Dengan Lensa Penyimpanan S3, Anda dapat mengidentifikasi jumlah byte unggahan multibagian yang tidak lengkap di akun Anda atau di seluruh organisasi Anda, termasuk unggahan multibagian yang tidak lengkap yang berusia lebih dari 7 hari. Untuk daftar lengkap metrik upload multibagian yang tidak lengkap, lihatGlosarium metrik Amazon S3 Storage Lens.

Sebagai praktik terbaik, kami menyarankan untuk mengonfigurasi aturan siklus hidup untuk kedaluwarsa unggahan multibagian yang tidak lengkap yang lebih lama dari jumlah hari tertentu. Saat Anda membuat aturan siklus hidup untuk kedaluwarsa unggahan multibagian yang tidak lengkap, sebaiknya 7 hari sebagai titik awal yang baik.

Langkah 1: Tinjau tren keseluruhan untuk unggahan multibagian yang tidak lengkap

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di panel navigasi kiri, pilihLensa Penyimpanan,Dasbor.
3. Di daftar Dasbor, pilih dasbor yang ingin Anda lihat.
4. Di dalamSnapshot untukkencanbagian, di bawahKategori metrik, pilihOptimalisasi biaya.

   YangSnapshot untukkencanpembaruan bagian untuk ditampilkanOptimalisasi biayametrik, yang meliputiByte unggahan multipart tidak lengkap yang berusia lebih dari 7 hari.

   Di bagan apa pun di dasbor Lensa Penyimpanan S3, Anda dapat melihat metrik untuk unggahan multibagian yang tidak lengkap. Anda dapat menggunakan metrik ini untuk menilai lebih lanjut dampak byte unggahan multibagian yang tidak lengkap pada penyimpanan Anda, termasuk kontribusinya terhadap tren pertumbuhan secara keseluruhan. Anda juga dapat menelusuri ke tingkat agregasi yang lebih dalam, menggunakanRekening,Wilayah AWS,Ember, atauKelas penyimpanantab untuk analisis data Anda yang lebih dalam. Sebagai contoh, lihat Temukan ember Amazon S3 yang dingin.

Langkah 2: Identifikasi bucket yang memiliki byte unggahan multibagian paling tidak lengkap tetapi tidak memiliki aturan siklus hidup untuk membatalkan unggahan multibagian yang tidak lengkap

Prasyarat

Untuk melihatBatalkan jumlah aturan siklus hidup unggahan multipart yang tidak lengkapmetrik di dasbor Lensa Penyimpanan S3 Anda, Anda harus mengaktifkan Lensa Penyimpanan S3Metrik dan rekomendasi tingkat lanjut, dan kemudian pilihMetrik pengoptimalan biaya tingkat lanjut. Untuk informasi selengkapnya, lihat Membuat dan memperbarui dasbor Lensa Penyimpanan Amazon S3.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihLensa Penyimpanan,Dasbor.
3. Di daftar Dasbor, pilih dasbor yang ingin Anda lihat.
4. Untuk mengidentifikasi bucket tertentu yang mengumpulkan unggahan multipart yang tidak lengkap yang berusia lebih dari 7 hari, bukaIkhtisar N teratas untukkencanbagian.

   Secara default,Ikhtisar N teratas untukkencanbagian menampilkan metrik untuk 3 bucket teratas. Anda dapat menambah atau mengurangi jumlah ember diAtas Nbidang. YangIkhtisar N teratas untukkencanbagian juga menunjukkan persentase perubahan dari hari sebelumnya atau minggu dan spark-line untuk memvisualisasikan tren. (Tren ini adalah tren 14 hari untuk metrik gratis dan tren 30 hari untuk metrik dan rekomendasi lanjutan.)

> **Note**
>
> Dengan metrik dan rekomendasi lanjutan Lensa Penyimpanan S3, metrik tersedia untuk kueri selama 15 bulan. Untuk informasi selengkapnya, lihat Pilihan metrik.

5. UntukMetrik, pilihByte unggahan multipart tidak lengkap yang berusia lebih dari 7 haridi dalamOptimalisasi biayakategori.

   Di bawahAtasnomorember, Anda dapat melihat ember dengan byte penyimpanan unggahan multipart paling tidak lengkap yang berusia lebih dari 7 hari.
6. Untuk melihat metrik tingkat bucket yang lebih rinci untuk unggahan multibagian yang tidak lengkap, gulir ke bagian atas halaman, lalu pilihEmbertab.
7. Gulir ke bawah keEmberbagian. UntukKategori metrik, pilihOptimalisasi biaya. Maka jelasRingkasan.

   YangEmberdaftar update untuk menampilkan semua yang tersediaOptimalisasi biayametrik untuk bucket yang ditampilkan.
8. Untuk menyaringEmberdaftar untuk menampilkan hanya metrik pengoptimalan biaya tertentu, pilih ikon preferensi
   ( ).
9. Hapus matikan untuk semua metrik pengoptimalan biaya hingga hanyaByte unggahan multipart tidak lengkap yang berusia lebih dari 7 haridanBatalkan jumlah aturan siklus hidup unggahan multipart yang tidak lengkaptetap dipilih.
10. (Opsional) Di bawahUkuran halaman, pilih jumlah ember untuk ditampilkan dalam daftar.
11. Pilih Konfirmasi.

    YangEmberdaftar pembaruan untuk menampilkan metrik tingkat ember untuk unggahan multibagian dan jumlah aturan siklus hidup yang tidak lengkap. Anda dapat menggunakan data ini untuk mengidentifikasi bucket yang memiliki byte unggahan multibagian paling tidak lengkap yang berusia lebih dari 7 hari dan tidak ada aturan siklus hidup untuk membatalkan upload multibagian yang tidak lengkap. Kemudian, Anda dapat menavigasi ke bucket ini di konsol S3 dan menambahkan aturan siklus hidup untuk menghapus unggahan multibagian yang tidak lengkap yang ditinggalkan.

Langkah 3: Tambahkan aturan siklus hidup untuk menghapus unggahan multibagian yang tidak lengkap setelah 7 hari

Untuk mengelola unggahan multibagian yang tidak lengkap secara otomatis, Anda dapat menggunakan konsol S3 untuk membuat konfigurasi siklus hidup untuk mengakhiri byte unggahan multibagian yang tidak lengkap dari bucket setelah jumlah hari tertentu. Untuk informasi selengkapnya, lihat Mengonfigurasi siklus hidup bucket untuk menghapus unggahan multibagian yang tidak lengkap.

Kurangi jumlah versi noncurrent yang dipertahankan

Saat diaktifkan, Versi S3 menyimpan beberapa salinan berbeda dari objek yang sama yang dapat Anda gunakan untuk memulihkan data dengan cepat jika objek dihapus atau ditimpa secara tidak sengaja. Jika Anda telah mengaktifkan Versi S3 tanpa mengonfigurasi aturan siklus hidup untuk transisi atau kedaluwarsa versi noncurrent, sejumlah besar versi noncurrent sebelumnya dapat terakumulasi, yang dapat memiliki implikasi biaya penyimpanan. Untuk informasi selengkapnya, lihat Menggunakan Penentuan Versi dalam bucket S3.

Langkah 1: Identifikasi bucket dengan versi objek paling noncurrent

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihLensa Penyimpanan,Dasbor.
3. Di daftar Dasbor, pilih dasbor yang ingin Anda lihat.
4. Di dalamSnapshot untukkencanbagian, di bawahKategori metrik, pilihOptimalisasi biaya.

   YangSnapshot untukkencanpembaruan bagian untuk ditampilkanOptimalisasi biayametrik, yang mencakup metrik untuk% byte versi noncurrent. Yang% byte versi noncurrentmetrik mewakili proporsi total byte penyimpanan Anda yang dikaitkan dengan versi noncurrent, dalam lingkup dasbor dan untuk tanggal yang dipilih.

   > **Note**
   >
   > Jika Anda% byte versi noncurrentlebih besar dari 10 persen penyimpanan Anda di tingkat akun, Anda mungkin menyimpan terlalu banyak versi objek.

5. Untuk mengidentifikasi ember tertentu yang mengumpulkan sejumlah besar versi noncurrent:

a. Gulir ke bawah kelkhtisar N teratas untukkencanbagian. UntukAtas N, masukkan jumlah ember yang ingin Anda lihat datanya.

b. UntukMetrik, pilih% byte versi noncurrent.

   Di bawahAtasnomorember, Anda dapat melihat ember (untuk nomor yang Anda tentukan) dengan yang tertinggi% byte versi noncurrent. YangIkhtisar N teratas untukkencanbagian juga menunjukkan persentase perubahan dari hari sebelumnya atau minggu dan spark-line untuk memvisualisasikan tren. Tren ini adalah tren 14 hari untuk metrik gratis dan tren 30 hari untuk metrik dan rekomendasi lanjutan.

   > **Note**
   >
   > Dengan metrik dan rekomendasi lanjutan Lensa Penyimpanan S3, metrik tersedia untuk kueri selama 15 bulan. Untuk informasi selengkapnya, lihat Pilihan metrik.

c. Untuk melihat metrik tingkat bucket yang lebih rinci untuk versi objek noncurrent, gulir ke bagian atas halaman, lalu pilihEmbertab.

   Dalam bagan atau visualisasi apa pun di dasbor S3 Storage Lens, Anda dapat menelusuri ke tingkat agregasi yang lebih dalam, menggunakanRekening,Wilayah AWS,Kelas penyimpanan, atauEmbertab. Sebagai contoh, lihat Temukan ember Amazon S3 yang dingin.

d. Di dalamEmberbagian, untukKategori metrik, pilihOptimalisasi biaya. maka jelaslahRingkasan.

   Anda sekarang dapat melihat% byte versi noncurrentmetrik, bersama dengan metrik lain yang terkait dengan versi noncurrent.

Langkah 2: Identifikasi bucket yang tidak memiliki aturan transisi dan siklus masa pakai kedaluwarsa untuk mengelola versi noncurrent

Prasyarat

Untuk melihatJumlah aturan siklus hidup transisi versi noncurrentdanJumlah aturan siklus hidup kedaluwarsa versi noncurrentmetrik di dasbor Lensa Penyimpanan S3 Anda, Anda harus mengaktifkan Lensa Penyimpanan S3Metrik dan rekomendasi tingkat lanjut, dan kemudian pilihMetrik

pengoptimalan biaya tingkat lanjut. Untuk informasi selengkapnya, lihat Membuat dan memperbarui dasbor Lensa Penyimpanan Amazon S3.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihLensa Penyimpanan,Dasbor.
3. Di daftar Dasbor, pilih dasbor yang ingin Anda lihat.
4. Di dasbor Lensa Penyimpanan, pilihEmbertab.
5. Gulir ke bawah keEmberbagian. UntukKategori metrik, pilihOptimalisasi biaya. Maka jelasRingkasan.

   YangEmberdaftar update untuk menampilkan semua yang tersediaOptimalisasi biayametrik untuk bucket yang ditampilkan.
6. Untuk menyaringEmberdaftar untuk menampilkan hanya metrik pengoptimalan biaya tertentu, pilih ikon preferensi
   ( ).
7. Kosongkan matikan untuk semua metrik pengoptimalan biaya hingga hanya hal-hal berikut yang tetap dipilih:
   - % byte versi noncurrent
   - Jumlah aturan siklus hidup transisi versi noncurrent
   - Jumlah aturan siklus hidup kedaluwarsa versi noncurrent
8. (Opsional) Di bawahUkuran halaman, pilih jumlah ember untuk ditampilkan dalam daftar.
9. Pilih Konfirmasi.

   YangEmberdaftar pembaruan untuk menampilkan metrik untuk byte versi noncurrent dan jumlah aturan siklus hidup versi noncurrent. Anda dapat menggunakan data ini untuk mengidentifikasi bucket yang memiliki persentase byte versi noncurrent yang tinggi tetapi tidak ada aturan siklus masa pakai transisi dan kedaluwarsa. Kemudian, Anda dapat menavigasi ke bucket ini di konsol S3 dan menambahkan aturan siklus hidup ke bucket ini.

Langkah 3: Tambahkan aturan siklus hidup untuk transisi atau berakhir versi objek noncurrent

Setelah menentukan bucket mana yang memerlukan penyelidikan lebih lanjut, Anda dapat menavigasi ke bucket dalam konsol S3 dan menambahkan aturan siklus hidup untuk kedaluwarsa

versi noncurrent setelah jumlah hari tertentu. Atau, untuk mengurangi biaya sambil tetap mempertahankan versi noncurrent, Anda dapat mengonfigurasi aturan siklus hidup untuk mentransisikan versi noncurrent ke salah satuAmazon S3 Glacierkelas penyimpanan. Untuk informasi selengkapnya, lihat Contoh 6: Menentukan aturan siklus hidup untuk bucket dengan dukungan Penentuan Versi.

Identifikasi bucket yang tidak memiliki aturan siklus hidup dan tinjau jumlah aturan siklus hidup

S3 Storage Lens menyediakan metrik jumlah aturan Siklus Hidup S3 yang dapat Anda gunakan untuk mengidentifikasi bucket yang tidak memiliki aturan siklus hidup. Untuk menemukan bucket yang tidak memiliki aturan siklus hidup, Anda dapat menggunakanTotal bucket tanpa aturan siklus hidupmetrik. Bucket tanpa konfigurasi Siklus Hidup S3 mungkin memiliki penyimpanan yang tidak lagi Anda butuhkan atau dapat bermigrasi ke kelas penyimpanan berbiaya lebih rendah. Anda juga dapat menggunakan metrik jumlah aturan siklus hidup untuk mengidentifikasi bucket yang kehilangan jenis aturan siklus hidup tertentu, seperti aturan kedaluwarsa atau transisi.

Prasyarat

Untuk melihat metrik jumlah aturan siklus hidup danTotal bucket tanpa aturan siklus hidupmetrik di dasbor Lensa Penyimpanan S3 Anda, Anda harus mengaktifkan Lensa Penyimpanan S3Metrik dan rekomendasi tingkat lanjut, dan kemudian pilihMetrik pengoptimalan biaya tingkat lanjut. Untuk informasi selengkapnya, lihat Membuat dan memperbarui dasbor Lensa Penyimpanan Amazon S3.

Langkah 1: Identifikasi bucket tanpa aturan siklus hidup

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihLensa Penyimpanan,Dasbor.
3. Di daftar Dasbor, pilih dasbor yang ingin Anda lihat.
4. Untuk mengidentifikasi bucket tertentu tanpa aturan siklus hidup, gulir ke bawah keIkhtisar N teratas untukkencanbagian.

   Secara default,Ikhtisar N teratas untukkencanbagian menampilkan metrik untuk 3 bucket teratas. DalamAtas Nbidang, Anda dapat meningkatkan jumlah ember. YangIkhtisar N teratas untukkencanbagian juga menunjukkan persentase perubahan dari hari sebelumnya atau minggu dan spark-line untuk memvisualisasikan tren. Tren ini adalah tren 14 hari untuk metrik gratis dan tren 30 hari untuk metrik dan rekomendasi lanjutan.

> **Note**
>
> Dengan metrik dan rekomendasi lanjutan Lensa Penyimpanan S3, metrik tersedia untuk kueri selama 15 bulan. Untuk informasi selengkapnya, lihat Pilihan metrik.

5. UntukMetrik, pilihTotal bucket tanpa aturan siklus hidupdariOptimalisasi biayakategori.
6. Tinjau data berikut untukTotal bucket tanpa aturan siklus hidup:

   - Atasnomorrekening- Lihat akun mana yang memiliki ember paling banyak tanpa aturan siklus hidup.
   - AtasnomorDaerah- Lihat rincian ember tanpa aturan siklus hidup berdasarkan Wilayah.
   - Atasnomorember- Lihat ember mana yang tidak memiliki aturan siklus hidup.

   Dalam bagan atau visualisasi apa pun di dasbor S3 Storage Lens, Anda dapat menelusuri ke tingkat agregasi yang lebih dalam, menggunakanRekening,Wilayah AWS,Kelas penyimpanan, atauEmbertab. Sebagai contoh, lihat Temukan ember Amazon S3 yang dingin.

   Setelah mengidentifikasi bucket mana yang tidak memiliki aturan siklus hidup, Anda juga dapat meninjau jumlah aturan siklus hidup tertentu untuk bucket Anda.

Langkah 2: Tinjau jumlah aturan siklus hidup untuk bucket Anda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihLensa Penyimpanan,Dasbor.
3. Di daftar Dasbor, pilih dasbor yang ingin Anda lihat.
4. Di dasbor Lensa Penyimpanan S3 Anda, pilihEmbertab.
5. Gulir ke bawah keEmberbagian. Di bawahKategori metrik, pilihOptimalisasi biaya. Maka jelasRingkasan.

   YangEmberdaftar update untuk menampilkan semua yang tersediaOptimalisasi biayametrik untuk bucket yang ditampilkan.
6. Untuk menyaringEmberdaftar untuk menampilkan hanya metrik pengoptimalan biaya tertentu, pilih ikon preferensi
   ( ).

7. Kosongkan matikan untuk semua metrik pengoptimalan biaya hingga hanya hal-hal berikut yang tetap dipilih:
   - Jumlah aturan siklus hidup transisi
   - Jumlah aturan siklus hidup kedaluwarsa
   - Jumlah aturan siklus hidup transisi versi noncurrent
   - Jumlah aturan siklus hidup kedaluwarsa versi noncurrent
   - Batalkan jumlah aturan siklus hidup unggahan multipart yang tidak lengkap
   - Jumlah aturan siklus hidup total
8. (Opsional) Di bawahUkuran halaman, pilih jumlah ember untuk ditampilkan dalam daftar.
9. Pilih Konfirmasi.

   YangEmberdaftar pembaruan untuk menampilkan metrik jumlah aturan siklus hidup untuk bucket Anda. Anda dapat menggunakan data ini untuk mengidentifikasi bucket tanpa aturan siklus hidup atau bucket yang kehilangan jenis aturan siklus hidup tertentu, misalnya aturan kedaluwarsa atau transisi. Kemudian, Anda dapat menavigasi ke bucket ini di konsol S3 dan menambahkan aturan siklus hidup ke bucket ini.

Langkah 3: Tambahkan aturan siklus hidup

Setelah mengidentifikasi bucket tanpa aturan siklus hidup, Anda dapat menambahkan aturan siklus hidup. Selengkapnya, lihat Menyetel konfigurasi siklus hidup pada bucket dan Contoh konfigurasi Siklus Hidup S3.

Menggunakan Lensa Penyimpanan S3 untuk melindungi data Anda

Anda dapat menggunakan metrik perlindungan data Amazon S3 Storage Lens untuk mengidentifikasi bucket di mana praktik terbaik perlindungan data belum diterapkan. Anda dapat menggunakan metrik ini untuk mengambil tindakan dan menerapkan pengaturan standar yang selaras dengan praktik terbaik untuk melindungi data Anda di seluruh bucket di akun atau organisasi Anda. Misalnya, Anda dapat menggunakan metrik perlindungan data untuk mengidentifikasi bucket yang tidak digunakanAWS Key Management Service(AWS KMS) kunci (SSE-KMS) untuk enkripsi default atau permintaan yang digunakanAWSTanda Tangan Versi 2 (SigV2).

Kasus penggunaan berikut menyediakan strategi untuk menggunakan dasbor Lensa Penyimpanan S3 Anda untuk mengidentifikasi outliers dan menerapkan praktik terbaik perlindungan data di seluruh bucket S3 Anda.

Topik

- Identifikasi bucket yang tidak menggunakan enkripsi sisi server denganAWS KMSuntuk enkripsi default (SSE-KMS)
- Identifikasi bucket yang mengaktifkan Versi S3
- Identifikasi permintaan yang digunakanAWSTanda Tangan Versi 2 (SigV2)
- Hitung jumlah total aturan replikasi untuk setiap bucket
- Identifikasi persentase byte Object Lock

Identifikasi bucket yang tidak menggunakan enkripsi sisi server denganAWS KMSuntuk enkripsi default (SSE-KMS)

Dengan enkripsi default Amazon S3, Anda dapat mengatur perilaku enkripsi default untuk bucket S3. Untuk informasi selengkapnya, lihat the section called “Mengatur enkripsi bucket default”.

Anda dapat menggunakanJumlah bucket yang diaktifkan SSE-KMSdan% SSE-KMS diaktifkan embermetrik untuk mengidentifikasi bucket yang menggunakan enkripsi sisi server denganAWS KMSkunci (SSE-KMS) untuk enkripsi default. S3 Storage Lens juga menyediakan metrik untuk byte tidak terenkripsi, objek tidak terenkripsi, byte terenkripsi, dan objek terenkripsi. Untuk daftar lengkap metrik, lihatGlosarium metrik Amazon S3 Storage Lens.

Anda dapat menganalisis metrik enkripsi SSE-KMS dalam konteks metrik enkripsi umum untuk mengidentifikasi bucket yang tidak menggunakan SSE-KMS. Jika Anda ingin menggunakan SSE-KMS untuk semua bucket di akun atau organisasi Anda, Anda kemudian dapat memperbarui pengaturan enkripsi default untuk bucket ini untuk menggunakan SSE-KMS. Selain SSE-KMS, Anda dapat menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3) atau kunci yang disediakan pelanggan (SSE-C). Untuk informasi selengkapnya, lihat Melindungi data dengan enkripsi.

Langkah 1: Identifikasi bucket mana yang menggunakan SSE-KMS untuk enkripsi default

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihLensa Penyimpanan,Dasbor.
3. DalamDasbordaftar, pilih nama dasbor yang ingin Anda lihat.
4. DalamTren dan distribusibagian, pilih% SSE-KMS mengaktifkan jumlah bucketuntuk metrik primer dan% byte terenkripsiuntuk metrik sekunder.

YangTren untukkencanupdate grafik untuk menampilkan tren untuk SSE-KMS dan byte terenkripsi.

5. Untuk melihat wawasan tingkat ember yang lebih terperinci untuk SSE-KMS:
   a. Pilih titik pada grafik. Sebuah kotak akan muncul dengan pilihan untuk wawasan yang lebih terperinci.
   b. PilihEmberdimensi. Lalu, pilih Terapkan.
6. DalamDistribusi oleh ember untukkencangrafik, pilihJumlah bucket yang diaktifkan SSE-KMSmetrik.
7. Anda sekarang dapat melihat ember mana yang mengaktifkan SSE-KMS dan mana yang tidak.

Langkah 2: Perbarui pengaturan enkripsi default bucket

Sekarang Anda telah menentukan bucket mana yang menggunakan SSE-KMS dalam konteks Anda% byte terenkripsi, Anda dapat mengidentifikasi bucket yang tidak menggunakan SSE-KMS. Anda kemudian dapat secara opsional menavigasi ke bucket ini dalam konsol S3 dan memperbarui pengaturan enkripsi default mereka untuk menggunakan SSE-KMS atau SSE-S3. Untuk informasi selengkapnya, lihat Mengonfigurasi enkripsi default.

Identifikasi bucket yang mengaktifkan Versi S3

Saat diaktifkan, fitur Versi S3 mempertahankan beberapa versi dari objek yang sama yang dapat digunakan untuk memulihkan data dengan cepat jika objek dihapus atau ditimpa secara tidak sengaja. Anda dapat menggunakanJumlah bucket yang diaktifkan versimetrik untuk melihat bucket mana yang menggunakan Versi S3. Kemudian, Anda dapat mengambil tindakan di konsol S3 untuk mengaktifkan Versi S3 untuk bucket lainnya.

Langkah 1: Identifikasi bucket yang mengaktifkan Versi S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi, pilih Storage Lens, Dasbor.
3. DalamDasbordaftar, pilih nama dasbor yang ingin Anda lihat.
4. DalamTren dan distribusibagian, pilihJumlah bucket yang diaktifkan versiuntuk metrik primer danEmberuntuk metrik sekunder.

YangTren untukkencanpembaruan bagan untuk menampilkan tren untuk bucket yang diaktifkan Versi S3. Tepat di bawah garis tren, Anda dapat melihatDistribusi kelas penyimpanandanDistribusi wilayahsubbagian.

5. Untuk melihat wawasan yang lebih terperinci untuk salah satu bucket yang Anda lihat diTren untukkencangrafik sehingga Anda dapat melakukan analisis yang lebih dalam, lakukan hal berikut:

   a. Pilih titik pada grafik. Sebuah kotak akan muncul dengan pilihan untuk wawasan yang lebih terperinci.

   b. Pilih dimensi untuk diterapkan ke data Anda untuk analisis lebih dalam:Akun,Wilayah AWS,Kelas penyimpanan, atauEmber. Lalu, pilih Terapkan.

6. Di dalamAnalisis gelembung oleh ember untukkencanbagian, pilihJumlah bucket yang diaktifkan versi,Ember, danEmber aktifmetrik.

   YangAnalisis gelembung oleh ember untukkencanbagian pembaruan untuk menampilkan data untuk metrik yang Anda pilih. Anda dapat menggunakan data ini untuk melihat bucket mana yang mengaktifkan Versi S3 dalam konteks jumlah total bucket Anda. Di dalamAnalisis gelembung oleh ember untukkencanbagian, Anda dapat memplot bucket Anda pada beberapa dimensi dengan menggunakan tiga metrik untuk mewakiliSumbu X,Sumbu Y, danUkurangelembung.

Langkah 2: Aktifkan Versi S3

Setelah mengidentifikasi bucket yang mengaktifkan Versi S3, Anda dapat mengidentifikasi bucket yang belum pernah mengaktifkan Versi S3 atau versi ditangguhkan. Kemudian, Anda dapat mengaktifkan versi untuk bucket ini di konsol S3. Untuk informasi selengkapnya, lihat Mengaktifkan Penentuan Versi pada bucket.

Identifikasi permintaan yang digunakanAWSTanda Tangan Versi 2 (SigV2)

Anda dapat menggunakanSemua permintaan tanda tangan yang tidak didukungmetrik untuk mengidentifikasi permintaan yang digunakanAWSTanda Tangan Versi 2 (SigV2). Data ini dapat membantu Anda mengidentifikasi aplikasi tertentu yang menggunakan SigV2. Anda kemudian dapat memigrasi aplikasi ini keAWSTanda Tangan Versi 4 (SiGv4).

SiGv4 adalah metode penandatanganan yang direkomendasikan untuk semua aplikasi S3 baru. SiGv4 memberikan peningkatan keamanan dan didukung di semuaWilayah AWS. Untuk informasi lebih lanjut, lihatPembaruan Amazon S3 - Periode pengusangan SigV2 diperpanjang & dimodifikasi.

Prasyarat

Untuk melihatSemua permintaan tanda tangan yang tidak didukungdi dasbor Lensa Penyimpanan S3 Anda, Anda harus mengaktifkan Lensa Penyimpanan S3Metrik dan rekomendasi lanjutandan kemudian pilihMetrik perlindungan data tingkat lanjut. Untuk informasi selengkapnya, lihat Membuat dan memperbarui dasbor Lensa Penyimpanan Amazon S3.

Langkah 1: Periksa SigV2 tren penandatanganan olehAkun AWS, Wilayah, dan ember

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihLensa Penyimpanan,Dasbor.
3. Di dalamDasbordaftar, pilih nama dasbor yang ingin Anda lihat.
4. Untuk mengidentifikasi bucket, akun, dan Wilayah tertentu dengan permintaan yang menggunakan SigV2:

   a. Di bawahIkhtisar N teratas untukkencan, diAtas N, masukkan jumlah ember yang ingin Anda lihat datanya.

   b. UntukMetrik, pilihSemua permintaan tanda tangan yang tidak didukungdariPerlindungan datakategori.

   YangIkhtisar N teratas untukkencanpembaruan untuk menampilkan data untuk permintaan SigV2 berdasarkan akun,Wilayah AWS, dan ember. YangIkhtisar N teratas untukkencanbagian juga menunjukkan persentase perubahan dari hari sebelumnya atau minggu dan spark-line untuk memvisualisasikan tren. Tren ini adalah tren 14 hari untuk metrik gratis dan tren 30 hari untuk metrik dan rekomendasi lanjutan.

   > **Note**
   >
   > Dengan metrik dan rekomendasi lanjutan Lensa Penyimpanan S3, metrik tersedia untuk kueri selama 15 bulan. Untuk informasi selengkapnya, lihat Pilihan metrik.

Langkah 2: Identifikasi bucket yang diakses oleh aplikasi melalui permintaan SigV2

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihLensa Penyimpanan,Dasbor.

3. Di dalamDasbordaftar, pilih nama dasbor yang ingin Anda lihat.
4. Di dasbor Lensa Penyimpanan, pilihEmbertab.
5. Gulir ke bawah keEmberbagian. Di bawahKategori metrik, pilihPerlindungan data. Maka jelasRingkasan.

   YangEmberdaftar update untuk menampilkan semua yang tersediaPerlindungan datametrik untuk bucket yang ditampilkan.
6. Untuk menyaringEmberdaftar untuk menampilkan hanya metrik perlindungan data tertentu, pilih ikon preferensi
   ( ).
7. Kosongkan matikan untuk semua metrik perlindungan data hingga hanya metrik berikut yang tetap dipilih:
   - Semua permintaan tanda tangan yang tidak didukung
   - % semua permintaan tanda tangan yang tidak didukung
8. (Opsional) Di bawahUkuran halaman, pilih jumlah ember untuk ditampilkan dalam daftar.
9. Pilih Konfirmasi.

   YangEmberdaftar pembaruan untuk menampilkan metrik tingkat ember untuk permintaan SigV2. Anda dapat menggunakan data ini untuk mengidentifikasi bucket tertentu yang memiliki permintaan SigV2. Kemudian, Anda dapat menggunakan informasi ini untuk memigrasi aplikasi Anda ke Sigv4. Untuk informasi lebih lanjut, lihat Permintaan Autentikasi (Tanda Tangan AWS Versi 4) di Referensi API Amazon Simple Storage Service.

Hitung jumlah total aturan replikasi untuk setiap bucket

Replikasi S3 memungkinkan penyalinan objek secara otomatis dan asinkron di seluruh bucket Amazon S3. Bucket yang dikonfigurasi untuk replikasi objek dapat dimiliki oleh Akun AWS yang sama atau oleh akun yang berbeda. Untuk informasi selengkapnya, lihat Mereplikasi ikhtisar objek.

Anda dapat menggunakan metrik jumlah aturan replikasi Lensa Penyimpanan S3 untuk mendapatkan informasi terperinci per bucket tentang bucket Anda yang dikonfigurasi untuk replikasi. Informasi ini mencakup aturan replikasi di dalam dan di seluruh bucket dan Regions.

Prasyarat

Untuk melihat metrik jumlah aturan replikasi di dasbor Lensa Penyimpanan S3, Anda harus mengaktifkan Lensa Penyimpanan S3Metrik dan rekomendasi lanjutandan kemudian pilihMetrik perlindungan data tingkat lanjut. Untuk informasi selengkapnya, lihat Membuat dan memperbarui dasbor Lensa Penyimpanan Amazon S3.

Langkah 1: Hitung jumlah total aturan replikasi untuk setiap bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihLensa Penyimpanan,Dasbor.
3. Di dalamDasbordaftar, pilih nama dasbor yang ingin Anda lihat.
4. Di dasbor Lensa Penyimpanan, pilihEmbertab.
5. Gulir ke bawah keEmberbagian. Di bawahKategori metrik, pilihPerlindungan data. Maka jelasRingkasan.
6. Untuk menyaringEmberdaftar untuk menampilkan metrik jumlah aturan replikasi saja, pilih ikon preferensi
   ( ).
7. Kosongkan matikan untuk semua metrik perlindungan data hingga hanya metrik penghitungan aturan replikasi yang tetap dipilih:
   - Jumlah aturan Replikasi Wilayah yang Sama
   - Jumlah aturan Replikasi Lintas Wilayah
   - Jumlah aturan replikasi akun yang sama
   - Jumlah aturan replikasi lintas akun
   - Jumlah aturan replikasi total
8. (Opsional) Di bawahUkuran halaman, pilih jumlah ember untuk ditampilkan dalam daftar.
9. Pilih Konfirmasi.

Langkah 2: Tambahkan aturan replikasi

Setelah Anda memiliki jumlah aturan replikasi per bucket, Anda dapat membuat aturan replikasi tambahan secara opsional. Untuk informasi selengkapnya, lihat Contoh untuk mengonfigurasi replikasi langsung.

Identifikasi persentase byte Object Lock

Dengan S3 Object Lock, Anda dapat menyimpan objek dengan menggunakanwrite-once-read-many(CACING)model. Anda dapat menggunakan Object Lock untuk membantu mencegah objek dihapus atau ditimpa untuk jumlah waktu yang tetap atau tanpa batas waktu. Anda dapat mengaktifkan Object Lock hanya ketika Anda membuat bucket dan juga mengaktifkan S3 Versioning. Namun, Anda dapat mengedit periode penyimpanan untuk versi objek individual atau menerapkan penahanan hukum untuk bucket yang mengaktifkan Kunci Objek. Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek S3.

Anda dapat menggunakan metrik Object Lock di S3 Storage Lens untuk melihat% Objek Lock bytemetrik untuk akun atau organisasi Anda. Anda dapat menggunakan informasi ini untuk mengidentifikasi bucket di akun atau organisasi Anda yang tidak mengikuti praktik terbaik perlindungan data Anda.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihLensa Penyimpanan,Dasbor.
3. Di dalamDasbordaftar, pilih nama dasbor yang ingin Anda lihat.
4. Di dalamSnapshotbagian, di bawahKategori metrik, pilihPerlindungan data.

   YangSnapshotpembaruan bagian untuk menampilkan metrik perlindungan data, termasuk% Objek Lock bytemetrik. Anda dapat melihat persentase keseluruhan byte Kunci Objek untuk akun atau organisasi Anda.
5. Untuk melihat% Objek Lock byteper ember, gulir ke bawah keIkhtisar Top Nbagian.

   Untuk mendapatkan data tingkat objek untuk Object Lock, Anda juga dapat menggunakanObjek Lock jumlah objekdan% Objek Kunci objekmetrik.
6. UntukMetrik, pilih% Objek Lock bytedariPerlindungan datakategori.

   Secara default,Ikhtisar N teratas untukkencanbagian menampilkan metrik untuk 3 bucket teratas. Di dalamAtas Nbidang, Anda dapat meningkatkan jumlah ember. YangIkhtisar N teratas untukkencanbagian juga menunjukkan persentase perubahan dari hari sebelumnya atau minggu dan spark-line untuk memvisualisasikan tren. Tren ini adalah tren 14 hari untuk metrik gratis dan tren 30 hari untuk metrik dan rekomendasi lanjutan.

> **Note**
>
> Dengan metrik dan rekomendasi lanjutan Lensa Penyimpanan S3, metrik tersedia untuk kueri selama 15 bulan. Untuk informasi selengkapnya, lihat Pilihan metrik.

7. Tinjau data berikut untuk% Objek Lock byte:

   - Atasnomorrekening- Lihat akun mana yang memiliki tertinggi dan terendah% Objek Lock byte.
   - AtasnomorDaerah- Lihat rincian% Objek Lock byteoleh Region.
   - Atasnomorember- Lihat ember mana yang memiliki tertinggi dan terendah% Objek Lock byte.

Menggunakan S3 Storage Lens untuk mengaudit pengaturan Object Ownership

Amazon S3 Object Ownership adalah pengaturan bucket S3 yang dapat Anda gunakan untuk menonaktifkan daftar kontrol akses (ACL) dan mengendalikan kepemilikan objek di ember Anda. Jika Anda menetapkan Kepemilikan Objek ke pemilik bucket yang diberlakukan, Anda dapat menonaktifkan daftar kontrol akses (ACL) dan mengambil kepemilikan setiap objek di bucket Anda. Pendekatan ini menyederhanakan manajemen akses untuk data yang disimpan di Amazon S3.

Secara default, ketika orang lainAkun AWS mengunggah objek ke bucket S3 Anda, akun tersebut (penulis objek) memiliki objek, memiliki akses ke sana, dan dapat memberi pengguna lain akses ke sana melalui ACL. Anda dapat menggunakan Kepemilikan Objek untuk mengubah perilaku default ini.

Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL. Oleh karena itu, kami menyarankan Anda menonaktifkan ACL, kecuali dalam keadaan yang tidak biasa di mana Anda harus mengontrol akses untuk setiap objek secara individual. Dengan menetapkan Kepemilikan Objek ke pemilik bucket yang diberlakukan, Anda dapat menonaktifkan ACL dan mengandalkan kebijakan untuk kontrol akses. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

Dengan metrik manajemen akses Lensa Penyimpanan S3, Anda dapat mengidentifikasi bucket yang tidak menonaktifkan ACL. Setelah mengidentifikasi bucket ini, Anda dapat memigrasi izin ACL ke kebijakan dan menonaktifkan ACL untuk bucket ini.

Topik

- Langkah 1: Identifikasi tren umum untuk pengaturan Kepemilikan Objek

- Langkah 2: Identifikasi tren tingkat ember untuk pengaturan Kepemilikan Objek
- Langkah 3: Perbarui pengaturan Kepemilikan Objek Anda ke pemilik bucket yang diberlakukan untuk menonaktifkan ACL

Langkah 1: Identifikasi tren umum untuk pengaturan Kepemilikan Objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Storage Lens, Dasbor.
3. Di daftar Dasbor, pilih nama dasbor yang ingin Anda lihat.
4. Di bagian Snapshot untuk tanggal, di bawah Kategori Metrik, pilih Manajemen akses.

   Bagian Snapshot untuk tanggal memperbarui untuk menampilkan metrik diberlakukan pemilik bucket% Object Ownership. Anda dapat melihat persentase keseluruhan bucket di akun atau organisasi Anda yang menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek untuk menonaktifkan ACL.

Langkah 2: Identifikasi tren tingkat ember untuk pengaturan Kepemilikan Objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Storage Lens, Dasbor.
3. Di daftar Dasbor, pilih nama dasbor yang ingin Anda lihat.
4. Untuk melihat metrik level bucket yang lebih detail, pilih tab Bucket.
5. Di bagian Distribusi berdasarkan bucket untuk tanggal, pilih metrik yang diberlakukan pemilik bucket% Object Ownership.

   Bagan update untuk menampilkan rincian per-bucket untuk pemilik bucket% Object Ownership diberlakukan. Anda dapat melihat bucket mana yang menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek untuk menonaktifkan ACL.
6. Untuk melihat pengaturan yang diberlakukan pemilik bucket dalam konteks, gulir ke bawah ke bagian Bucket. Untuk kategori Metrik, pilih Manajemen akses. Kemudian jelas Ringkasan.

   Daftar Bucket menampilkan data untuk ketiga pengaturan Kepemilikan Objek: pemilik bucket yang diberlakukan, pilihan pemilik bucket, dan penulis objek.

7. Untuk memfilter daftar Bucket agar menampilkan metrik hanya untuk setelan Kepemilikan Objek tertentu, pilih ikon preferensi ( ).
8. Kosongkan metrik yang tidak ingin Anda lihat.
9. (Opsional) Di bawah Ukuran halaman, pilih jumlah bucket yang akan ditampilkan dalam daftar.
10. Pilih Konfirmasi.

Langkah 3: Perbarui pengaturan Kepemilikan Objek Anda ke pemilik bucket yang diberlakukan untuk menonaktifkan ACL

Setelah mengidentifikasi bucket yang menggunakan pengaturan pilihan penulis objek dan pemilik bucket untuk Kepemilikan Objek, Anda dapat memigrasi izin ACL ke kebijakan bucket. Setelah selesai memigrasi izin ACL, Anda kemudian dapat memperbarui pengaturan Kepemilikan Objek ke pemilik bucket yang diberlakukan untuk menonaktifkan ACL. Untuk informasi selengkapnya, lihat Prasyarat untuk menonaktifkan ACL.

Menggunakan metrik S3 Storage Lens untuk meningkatkan kinerja

Jika Anda mengaktifkan metrik lanjutan Lensa Penyimpanan S3, Anda dapat menggunakan metrik kode status terperinci untuk mendapatkan jumlah permintaan yang berhasil atau gagal. Anda dapat menggunakan informasi ini untuk memecahkan masalah akses atau kinerja. Metrik kode status terperinci menunjukkan jumlah kode status HTTP, seperti 403 Forbidden dan 503 Service Unavailable. Anda dapat memeriksa tren keseluruhan untuk metrik kode status terperinci di seluruh bucket, akun, dan organisasi S3. Kemudian, Anda dapat menelusuri metrik tingkat ember untuk mengidentifikasi beban kerja yang saat ini mengakses bucket ini dan menyebabkan kesalahan.

Misalnya, Anda dapat melihat metrik 403 Forbidden error count untuk mengidentifikasi beban kerja yang mengakses bucket tanpa izin yang benar diterapkan. Setelah mengidentifikasi beban kerja ini, Anda dapat melakukan penyelaman jauh di luar S3 Storage Lens untuk memecahkan masalah 403 kesalahan Terlarang Anda.

Contoh ini menunjukkan cara melakukan analisis tren untuk kesalahan 403 Forbidden dengan menggunakan 403 Forbidden error count dan% 403 Forbidden error metrics. Anda dapat menggunakan metrik ini untuk mengidentifikasi beban kerja yang mengakses bucket tanpa izin yang benar diterapkan. Anda dapat melakukan analisis tren serupa untuk metrik kode status terperinci lainnya. Untuk informasi selengkapnya, lihat Glosarium metrik Amazon S3 Storage Lens.

Prasyarat

Untuk melihat metrik kode status terperinci di dasbor Lensa Penyimpanan S3, Anda harus mengaktifkan metrik dan rekomendasi Lanjutan Lensa Penyimpanan S3, lalu pilih Metrik kode status terperinci. Untuk informasi selengkapnya, lihat Membuat dan memperbarui dasbor Lensa Penyimpanan Amazon S3.

Topik

- Langkah 1: Lakukan analisis tren untuk kode status HTTP individual
- Langkah 2: Menganalisis jumlah kesalahan dengan ember
- Langkah 3: Memecahkan masalah

Langkah 1: Lakukan analisis tren untuk kode status HTTP individual

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Storage Lens (Dasbor).
3. Di daftar Dasbor, pilih nama dasbor yang ingin Anda lihat.
4. Di bagian Tren dan distribusi, untuk metrik Primer, pilih 403 Jumlah kesalahan Terlarang dari kategori Kode status terperinci. Untuk Metrik sekunder, pilih% 403 Kesalahan terlarang.
5. Gulir ke bawah ke bagian Ikhtisar N Atas untuk tanggal. Untuk Metrik, pilih 403 Jumlah kesalahan Terlarang atau% 403 Error terlarang dari kategori Kode status terperinci.

   Ringkasan N atas untuk tanggal bagian update untuk menampilkan atas 403 Terlarang kesalahan menghitung berdasarkan akun,Wilayah AWS, dan ember.

Langkah 2: Menganalisis jumlah kesalahan dengan ember

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Storage Lens (Dasbor).
3. Di daftar Dasbor, pilih nama dasbor yang ingin Anda lihat.
4. Di dasbor Lensa Penyimpanan, pilih tab Bucket.
5. Gulir ke bawah hingga bagian ember. Untuk kategori Metrik, pilih Metrik kode status terperinci. Kemudian jelas Ringkasan.

Daftar Bucket diperbarui untuk menampilkan semua metrik kode status terperinci yang tersedia. Anda dapat menggunakan informasi ini untuk melihat bucket mana yang memiliki sebagian besar kode status HTTP tertentu dan kode status mana yang umum di seluruh bucket.

6. Untuk memfilter daftar Bucket agar hanya menampilkan metrik kode status terperinci tertentu, pilih ikon preferensi ( ).
7. Kosongkan matikan untuk metrik kode status terperinci yang tidak ingin Anda lihat di daftar Bucket.
8. (Opsional) Di bawah Ukuran halaman, pilih jumlah bucket yang akan ditampilkan dalam daftar.
9. Pilih Konfirmasi.

   Daftar Bucket menampilkan metrik jumlah kesalahan untuk jumlah bucket yang Anda tentukan. Anda dapat menggunakan informasi ini untuk mengidentifikasi bucket tertentu yang mengalami banyak kesalahan dan memecahkan masalah kesalahan berdasarkan bucket.

Langkah 3: Memecahkan masalah

Setelah mengidentifikasi bucket dengan proporsi kode status HTTP tertentu yang tinggi, Anda dapat memecahkan masalah kesalahan ini. Untuk informasi selengkapnya, lihat yang berikut:

- Mengapa saya mendapatkan kesalahan 403 Forbidden ketika saya mencoba mengunggah file di Amazon S3?
- Mengapa saya mendapatkan error 403 Forbidden saat mencoba memodifikasi kebijakan bucket di Amazon S3?
- Bagaimana cara memecahkan masalah 403 kesalahan Terlarang dari bucket Amazon S3 saya di mana semua sumber daya berasal dari yang samaAkun AWS?
- Bagaimana cara memecahkan masalah kesalahan HTTP 500 atau 503 dari Amazon S3?

Glosarium metrik Amazon S3 Storage Lens

Glosarium metrik Lensa Penyimpanan Amazon S3 menyediakan daftar lengkap metrik gratis dan canggih untuk Lensa Penyimpanan S3.

S3 Storage Lens menawarkan metrik gratis untuk semua dasbor dan konfigurasi, dengan opsi untuk meningkatkan ke metrik lanjutan.

- Metrik gratis berisi metrik yang relevan dengan penggunaan penyimpanan Anda, seperti jumlah bucket dan objek di akun Anda. Metrik gratis juga mencakup metrik berbasis kasus penggunaan, seperti pengoptimalan biaya dan metrik perlindungan data. Semua metrik gratis dikumpulkan setiap hari, dan data tersedia untuk kueri hingga 14 hari.
- Metrik dan rekomendasi lanjutan mencakup semua metrik dalam metrik gratis bersama dengan metrik tambahan, seperti perlindungan data tingkat lanjut dan metrik pengoptimalan biaya. Metrik lanjutan juga mencakup kategori metrik tambahan, seperti metrik aktivitas dan metrik kode status terperinci. Data metrik lanjutan tersedia untuk kueri selama 15 bulan.

  Ada biaya tambahan ketika Anda menggunakan S3 Storage Lens dengan metrik dan rekomendasi tingkat lanjut. Untuk informasi selengkapnya, lihat Harga Amazon S3. Untuk informasi selengkapnya tentang metrik lanjutan dan fitur rekomendasi, lihatPilihan metrik.

> **Note**
>
> Untuk grup Lensa Penyimpanan, hanya metrik penyimpanan tingkat gratis yang tersedia. Metrik tingkat lanjut tidak tersedia di tingkat grup Lensa Penyimpanan.

Nama metrik

Kolom nama Metrik dalam tabel berikut memberikan nama masing-masing Lensa Penyimpanan S3 di konsol S3. Kolom CloudWatch dan ekspor memberikan nama setiap metrik di Amazon CloudWatch dan file ekspor metrik yang dapat Anda konfigurasikan di dasbor Lensa Penyimpanan S3 Anda.

Rumus metrik turunan

Metrik turunan tidak tersedia untuk ekspor metrik dan opsi CloudWatch penerbitan. Namun, Anda dapat menggunakan rumus metrik yang ditampilkan di kolom Rumus metrik turunan untuk menghitungnya.

Menafsirkan simbol awalan Lensa Penyimpanan Amazon S3 untuk kelipatan unit metrik (K, M, G, dan sebagainya)

Kelipatan unit metrik Lensa Penyimpanan S3 ditulis dengan simbol awalan. Simbol awalan ini cocok dengan simbol Sistem Satuan Internasional (SI) yang distandarisasi oleh International Bureau of Weights and Measures (BIPM). Simbol-simbol ini juga digunakan dalam Unified Code for Units of Measure (UCUM). Untuk informasi selengkapnya, lihat Daftar simbol awalan SI.

> **Note**
>
> - Unit pengukuran untuk byte penyimpanan S3 adalah dalam gigabyte biner (GB), di mana 1 GB adalah 2^{30} byte, 1 TB adalah 2^{40} byte, dan 1 PB adalah 2^{50} byte. Unit pengukuran ini juga dikenal sebagai gibibyte (GiB), sebagaimana didefinisikan oleh International Electrotechnical Commission (IEC).
> - Ketika sebuah objek mencapai akhir masa pakainya berdasarkan konfigurasi siklus hidupnya, Amazon S3 mengantri objek untuk dihapus dan menghapusnya secara asinkron. Oleh karena itu, mungkin ada penundaan antara tanggal kedaluwarsa dan tanggal ketika Amazon S3 menghapus objek. Lensa Penyimpanan S3 tidak menyertakan metrik untuk objek yang telah kedaluwarsa tetapi belum dihapus. Untuk informasi selengkapnya tentang tindakan kedaluwarsa di Siklus Hidup S3, lihat. Mengakhiri objek

Glosarium metrik Lensa Penyimpanan S3

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingl 1	Kate 2	Be	Form metri turur
Penyimpanan total	StorageBytes	Total penyimpan an, termasuk unggahan multibagian yang tidak lengkap, metadata objek, dan penanda hapus	Grati	Ring	N	-
Hitungan objek	ObjectCount	Jumlah objek total	Grati	Ring	N	-
Ukuran objek rata-rata	-	Ukuran objek rata-rata	Grati	Ring	Y	jumlah (StorageBytes) / jumlah () ObjectCount

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Ember aktif	-	Jumlah total bucket dalam penggunaan aktif dengan penyimpanan > 0 byte	Grati	Ring	Y	-
Bucket	-	Jumlah total ember	Grati	Ring	Y	-
Akun	-	Jumlah akun yang penyimpanannya dalam cakupan	Grati	Ring	Y	-
Byte versi saat ini	CurrentVersionStorageBytes	Jumlah byte yang merupakan versi objek saat ini	Grati	Optiasi biaya	N	-
% byte versi saat ini	-	Persentase byte dalam lingkup yang merupakan versi objek saat ini	Grati	Optiasi biaya	Y	jumlah (CurrentVersionStorageBytes) / jumlah () StorageBytes
Jumlah objek versi saat ini	CurrentVersionObjectCount	Hitungan objek versi saat ini	Grati	Optiasi biaya	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% objek versi saat ini	-	Persentase objek dalam lingkup yang merupakan versi saat ini	Grati	Optim asi biaya	Y	jumlah (CurrentVersionObjectCount) / jumlah () ObjectCount
Byte versi tidak saat ini	NonCurrentVersionStorageBytes	Jumlah byte versi noncurrent	Grati	Optim asi biaya	N	-
% byte versi noncurrent	-	Persentase byte dalam lingkup yang merupakan versi noncurrent	Grati	Optim asi biaya	Y	jumlah (NonCurrentVersionStorageBytes) / jumlah () StorageBytes
Jumlah objek versi tidak saat ini	NonCurrentVersionObjectCount	Hitungan versi objek noncurrent	Grati	Optim asi biaya	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% objek versi noncurrent	-	Persentase objek dalam cakupan yang merupakan versi tidak terkini	Grati	Optim asi biaya	Y	jumlah (NonCurrentVersionObjectCount) / jumlah () ObjectCount
Hapus byte penanda	DeleteMarkerStorageBytes	Jumlah byte dalam lingkup yang menghapus penanda	Grati	Optim asi biaya	N	-
% hapus byte penanda	-	Persentase byte dalam lingkup yang menghapus penanda	Grati	Optim asi biaya	Y	jumlah (DeleteMarkerStorageBytes) / jumlah () StorageBytes
Hapus jumlah objek penanda	DeleteMarkerObjectCount	Jumlah total objek dengan delete marker	Grati	Optim asi biaya	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% hapus objek penanda	-	Persentase objek dalam cakupan dengan delete marker	Grati	Optim asi biaya	Y	jumlah (DeleteMarkerObjectCount) / jumlah () ObjectCount
Byte unggahan multipart yang tidak lengkap	IncompleteMultipartUploadStorageBytes	Total byte dalam lingkup untuk unggahan multipart yang tidak lengkap	Grati	Optim asi biaya	N	-
% byte unggahan multipart tidak lengkap	-	Persentase byte dalam lingkup yang merupakan hasil dari unggahan multipart yang tidak lengkap	Grati	Optim asi biaya	Y	jumlah (IncompleteMultipartUploadStorageBytes) / jumlah () StorageBytes
Jumlah objek unggahan multipart yang tidak lengkap	IncompleteMultipartUploadObjectCount	Jumlah objek dalam cakupan yang merupakan unggahan multipart yang tidak lengkap	Grati	Optim asi biaya	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% objek unggahan multipart tidak lengkap	-	Persentase objek dalam cakupan yang merupakan unggahan multipart yang tidak lengkap	Grati	Optim asi biaya	Y	jumlah (IncompleteMultipartUploadObjectCount) / jumlah () ObjectCount
Byte penyimpanan unggahan multipart yang tidak lengkap lebih dari 7 hari	Tidak StorageBytesOlderThanLengkapu7Days	Total byte dalam lingkup untuk unggahan multipart yang tidak lengkap yang berusia lebih dari 7 hari	Grati	Optim asi biaya	N	-
% byte penyimpanan unggahan multipart tidak lengkap lebih dari 7 hari	-	Persentase byte untuk unggahan multipart yang tidak lengkap yang berusia lebih dari 7 hari	Grati	Optim asi biaya	Y	jumlah (tidak lengkapuStorageBytesOlderThan7Hari) / sum () StorageBytes

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Jumlah objek unggahan multipart yang tidak lengkap lebih dari 7 hari	Tidak ObjectCountOlderThan Lengkapu 7Days	Jumlah objek yang belum selesai diunggah multipart berusia lebih dari 7 hari	Grati	Optimasi biaya	N	-
% jumlah objek unggahan multipart tidak lengkap lebih dari 7 hari	-	Persentase objek yang tidak lengkap diunggah multipart berusia lebih dari 7 hari	Grati	Optimasi biaya	Y	jumlah (tidak lengkapu ObjectCountOlderThan 7Hari) / sum () ObjectCount
Jumlah aturan siklus hidup transisi	TransitionLifecycleRuleCount	Hitungan aturan siklus hidup untuk mentransisikan objek ke kelas penyimpanan lain	Lanju	Optimasi biaya	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Aturan siklus hidup transisi rata-rata per ember	-	Jumlah rata-rata aturan siklus hidup untuk mengalihkan objek ke kelas penyimpanan lain	Lanju	Optim asi biaya	Y	jumlah (TransitionLifecycleRuleCount) / jumlah () DistinctNumberOfBuckets
Jumlah aturan siklus hidup kedaluwarsa	ExpirationLifecycleRuleCount	Hitungan aturan siklus hidup untuk objek kedaluwarsa	Lanju	Optim asi biaya	N	-
Aturan siklus hidup kedaluwarsa rata-rata per ember	-	Jumlah rata-rata aturan siklus hidup untuk objek kedaluwarsa	Lanju	Optim asi biaya	Y	jumlah (ExpirationLifecycleRuleCount) / jumlah () DistinctNumberOfBuckets
Jumlah aturan siklus hidup transisi versi tidak saat ini	NoncurrentVersionTransitionLifecycleRuleCount	Hitungan aturan siklus hidup untuk mentransisikan versi objek noncurrent ke kelas penyimpanan lain	Lanju	Optim asi biaya	N	

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Rata-rata aturan siklus hidup transisi versi noncurrent per bucket	-	Jumlah rata-rata aturan siklus hidup untuk mentransisikan versi objek noncurrent ke kelas penyimpanan lain	Lanju	Optimasi biaya	Y	jumlah (NoncurrentVersionTransitionLifecycleRuleCount) / jumlah () DistinctNumberOfBuckets
Jumlah aturan siklus hidup kedaluwarsa versi tidak saat ini	NoncurrentVersionExpirationLifecycleRuleCount	Hitungan aturan siklus hidup untuk kedaluwarsa versi objek noncurrent	Lanju	Optimasi biaya	N	-
Rata-rata aturan siklus hidup kedaluwarsa versi noncurrent per bucket	-	Jumlah rata-rata aturan siklus hidup yang kedaluwarsa versi objek noncurrent	Lanju	Optimasi biaya	Y	jumlah (NoncurrentVersionExpirationLifecycleRuleCount) / jumlah () DistinctNumberOfBuckets

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Batalkan jumlah aturan siklus hidup unggahan multibagian yang tidak lengkap	AbortIncompleteMPULifecycleRuleCount	Hitungan aturan siklus hidup untuk menghapus unggahan multibagian yang tidak lengkap	Lanju	Optimasi biaya	N	-
Rata-rata membatalkan aturan siklus hidup unggahan multibagian yang tidak lengkap per ember	-	Jumlah rata-rata aturan siklus hidup untuk menghapus unggahan multibagian yang tidak lengkap	Lanju	Optimasi biaya	Y	jumlah (AbortIncompleteMPULifecycleRuleCount) / jumlah () DistinctNumberOfBuckets
Jumlah aturan siklus hidup penanda hapus objek kedaluwarsa	ExpiredObjectDeleteMarkerLifecycleRuleCount	Hitungan aturan siklus hidup untuk menghapus penanda penghapusan objek kedaluwarsa	Lanju	Optimasi biaya	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Rata-rata aturan siklus hidup penanda hapus objek kedaluwarsa per ember	-	Jumlah rata-rata aturan siklus hidup untuk menghapus penanda penghapusan objek kedaluwarsa	Lanju	Optim asi biaya	Y	jumlah (ExpiredObjectDeleteMarkerLifecycleRuleCount) / jumlah () DistinctNumberOfBuckets
Jumlah aturan siklus hidup total	TotalLifecycleRuleCount	Jumlah total aturan siklus hidup	Lanju	Optim asi biaya	N	-
Jumlah aturan siklus hidup rata-rata per ember	-	Jumlah rata-rata aturan siklus hidup	Lanju	Optim asi biaya	Y	jumlah (TotalLifecycleRuleCount) / jumlah () DistinctNumberOfBuckets
Byte terenkripsi	EncryptedStorageBytes	Jumlah total byte terenkripsi	Grati	Perlin gan data	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% byte terenkripsi	-	Persentase total byte yang dienkripsi	Grati	Perli gan data	Y	jumlah (EncryptedObjectCount) / jumlah () StorageBytes
Jumlah objek terenkripsi	EncryptedObjectCount	Jumlah total objek yang dienkripsi	Grati	Perli gan data	N	-
% objek terenkripsi	-	Persentase objek yang dienkripsi	Grati	Perli gan data	Y	jumlah (EncryptedStorageBytes) / jumlah () ObjectCount
Byte tidak terenkripsi	UnencryptedStorageBytes	Jumlah byte yang tidak dienkripsi	Grati	Perli gan data	Y	jumlah (StorageBytes) - jumlah (EncryptedStorageBytes)

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% byte tidak terenkripsi	-	Persentase byte yang tidak terenkripsi	Grati	Perli gan data	Y	jumlah (UnencryptedStorageBytes) / jumlah () StorageBytes
Jumlah objek yang tidak terenkripsi	UnencryptedObjectCount	Jumlah total objek yang tidak terenkripsi	Grati	Perli gan data	Y	jumlah (ObjectCount) - jumlah (EncryptedObjectCount)
% objek tidak terenkripsi	-	Persentase objek yang tidak terenkripsi	Grati	Perli gan data	Y	jumlah (UnencryptedStorageBytes) / jumlah () ObjectCount
Sumber byte penyimpanan yang direplikasi	ReplicatedStorageBytesSource	Jumlah total byte yang direplikasi dari bucket sumber	Grati	Perli gan data	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% sumber byte yang direplikasi	-	Persentase total byte yang direplikasi dari bucket sumber	Grati	Perlingan data	Y	jumlah (ReplicatedStorageBytesSource) / jumlah () StorageBytes
Sumber jumlah objek yang direplikasi	ReplicatedObjectCountSource	Hitungan objek yang direplikasi dari ember sumber	Grati	Perlingan data	N	-
% sumber objek yang direplikasi	-	Persentase total objek yang direplikasi dari ember sumber	Grati	Perlingan data	Y	jumlah (ReplicatedStorageObjectCount) / jumlah () ObjectCount
Tujuan byte penyimpanan replikasi	ReplicatedStorageBytes	Jumlah total byte yang direplikasi ke bucket tujuan	Grati	Perlingan data	Y	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% tujuan byte direplikasi	-	Persentase total byte yang direplikasi ke bucket tujuan	Grati	Perlig an data	Y	jumlah (ReplicatedStorageBytes) / jumlah () StorageBytes
Tujuan jumlah objek yang direplikasi	ReplicatedObjectCount	Hitungan objek yang direplikasi ke ember tujuan	Grati	Perlig an data	Y	-
% objek tujuan direplikasi	-	Persentase total objek yang direplikasi ke bucket tujuan	Grati	Perlig an data	Y	jumlah (ReplicatedObjectCount) / jumlah () ObjectCount

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Byte Kunci Objek	ObjectLockEnabledStorageBytes	Jumlah total byte penyimpanan yang diaktifkan Object Lock	Grati	Perligan data	Y	jumlah (UnencryptedStorageBytes) / sum (ObjectLockEnabledStorageCount) - sum () ObjectLockEnabledStorageBytes
% Byte Kunci Objek	-	Persentase byte penyimpanan yang diaktifkan Object Lock	Grati	Perligan data	Y	jumlah (ObjectLockEnabledStorageBytes) / jumlah () StorageBytes
Object Lock jumlah objek	ObjectLockEnabledObjectCount	Jumlah total objek Object Lock	Grati	Perligan data	Y	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% Objek Kunci Objek	-	Persentase total objek yang mengaktifkan Object Lock	Grati	Perli gan data	Y	jumlah (ObjectLo ckEnabled ObjectCou nt) / jumlah () ObjectCou nt
Jumlah bucket berkemampuan versi	VersioningEnabledBucketCount	Hitungan bucket yang mengaktifkan Versi S3	Grati	Perli gan data	N	-
% ember berkemampuan versi	-	Persentase bucket yang mengaktifkan Versi S3	Grati	Perli gan data	Y	jumlah (VersioningEnabledBucketCount) / jumlah () DistinctNumberOfBuckets
Jumlah bucket yang diaktifkan penghapusan MFA	MFA DeleteEnabledBucketCount	Hitungan bucket yang memiliki penghapusan MFA (multi-faktor otentikasi) diaktifkan	Grati	Perli gan data	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% ember yang diaktifkan penghapusan MFA	-	Persentase bucket yang memiliki penghapusan MFA (multi-faktor otentikasi) diaktifkan	Grati	Perlu gan data	Y	jumlah (MFADeleteEnabledBucketCount) / jumlah () DistinctNumberOfBuckets
Jumlah bucket yang diaktifkan SSE-KMS	SSEKMEnabledBucketCount	Jumlah bucket yang menggunakan enkripsi sisi server dengan AWS Key Management Service kunci (SSE-KMS) untuk enkripsi bucket default	Grati	Perlu gan data	N	-
% SSE-KMS mengaktifkan bucket	-	Persentase bucket yang SSE-KMS untuk enkripsi bucket default	Grati	Perlu gan data	Y	jumlah (SSEKMSEnabledBucketCount) / jumlah () DistinctNumberOfBuckets

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Semua permintaan tanda tangan yang tidak didukung	AllUnsupportedSignatureRequests	Jumlah total permintaan yang menggunakan versi AWS tanda tangan yang tidak didukung	Lanju	Perlingan data	N	-
% semua permintaan tanda tangan yang tidak didukung	-	Persentase permintaan yang menggunakan versi AWS tanda tangan yang tidak didukung	Lanju	Perlingan data	Y	jumlah (AllUnsupportedSignatureRequests) / jumlah () AllRequests
Semua permintaan TLS yang tidak didukung	AllUnsupportedTLSRequests	Jumlah permintaan yang menggunakan versi Transport Layer Security (TLS) yang tidak didukung	Lanju	Perlingan data	N	-
% semua permintaan TLS yang tidak didukung	-	Persentase permintaan yang menggunakan versi TLS yang tidak didukung	Lanju	Perlingan data	Y	jumlah (AllUnsupportedtlsRequests) / sum () AllRequests

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Semua permintaan SSE-KMS	AllsSekms Requests	Jumlah total permintaan yang menentukan SSE-KMS	Lanju	Perli gan data	N	-
% semua permintaan SSE-KMS	-	Persentase permintaan yang menentukan SSE-KMS	Lanju	Perli gan data	Y	jumlah (allsSekm sRequests) / sum () AllReques ts
Jumlah aturan Replikasi Wilayah yang Sama	SameRegio nReplicat ionRuleCount	Hitungan aturan replikasi untuk Same-Region Replication (SRR)	Lanju	Perli gan data	N	-
Aturan Replikasi Wilayah Sama Rata-rata per ember	-	Jumlah rata-rata aturan replikasi untuk SRR	Lanju	Perli gan data	Y	jumlah (SameRegi onReplica tionRuleC ount) / jumlah () DistinctN umberOfBu ckets

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Jumlah aturan Replikasi Lintas Wilayah	CrossRegionReplicationRuleCount	Hitungan aturan replikasi untuk Replikasi Lintas Wilayah (CRR)	Lanju	Perligan data	N	-
Aturan Replikasi Lintas Wilayah Rata-rata per ember	-	Jumlah rata-rata aturan replikasi untuk CRR	Lanju	Perligan data	Y	jumlah (CrossRegionReplicationRuleCount) / jumlah () DistinctNumberOfBuckets
Jumlah aturan replikasi akun yang sama	SameAccountReplicationRuleCount	Hitungan aturan replikasi untuk replikasi dalam akun yang sama	Lanju	Perligan data	N	-
Aturan replikasi akun yang sama rata-rata per bucket	-	Jumlah rata-rata aturan replikasi untuk replikasi dalam akun yang sama	Lanju	Perligan data	Y	jumlah (SameAccountReplicationRuleCount) / jumlah () DistinctNumberOfBuckets

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Hitungan aturan replikasi lintas akun	CrossAccountReplicationRuleCount	Hitungan aturan replikasi untuk replikasi lintas akun	Lanju	Perlingan data	N	-
Aturan replikasi lintas akun rata-rata per bucket	-	Jumlah rata-rata aturan replikasi untuk replikasi lintas akun	Lanju	Perlingan data	Y	jumlah (CrossAccountReplicationRuleCount) / jumlah () DistinctNumberOfBuckets
Jumlah aturan replikasi tujuan tidak valid	InvalidDestinationReplicationRuleCount	Hitungan aturan replikasi dengan tujuan replikasi yang tidak valid	Lanju	Perlingan data	N	-
Rata-rata aturan replikasi tujuan tidak valid per bucket	-	Jumlah rata-rata aturan replikasi dengan tujuan replikasi yang tidak valid	Lanju	Perlingan data	Y	jumlah (InvalidReplicationRuleCount) / jumlah () DistinctNumberOfBuckets

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Jumlah aturan replikasi total	-	Jumlah aturan replikasi total	Lanju	Perli gan data	Y	-
Jumlah aturan replikasi rata-rata per ember	-	Jumlah aturan replikasi total rata-rata	Lanju	Perli gan data	Y	sum (semua metrik hitungan aturan replikasi) / sum () DistinctNumberOfBuckets
Pemilik bucket Kepemilikan Objek memberlakukan hitungan ember	ObjectOwnershipBucketOwnerEnforcedBucketCount	Jumlah total bucket yang memiliki daftar kontrol akses (ACL) dinonaktifkan dengan menggunakan setelan yang diberlakukan pemilik bucket untuk Kepemilikan Objek	Grati	Man akse	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% Object Ownership bucket bucket diberlakukan bucket	-	Persentase bucket yang menonaktifkan ACL dengan menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek	Grati	Man akse	Y	jumlah (ObjectOwnershipBucketOwnerEnforcedBucketCount) / jumlah () DistinctNumberOfBuckets
Pemilik bucket Kepemilikan Objek lebih disukai jumlah bucket	ObjectOwnershipBucketOwnerPreferredBucketCount	Jumlah total bucket yang menggunakan pengaturan pilihan pemilik bucket untuk Object Ownership	Grati	Man akse	N	-
% Object Ownership bucket bucket pilihan bucket bucket	-	Persentase bucket yang menggunakan pengaturan pilihan pemilik bucket untuk Kepemilikan Objek	Grati	Man akse	Y	jumlah (ObjectOwnershipBucketOwnerPreferredBucketCount) / jumlah () DistinctNumberOfBuckets

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Jumlah ember penulis objek Kepemilikan Objek	ObjectOwnershipObjectWriterBucketCount	Jumlah total bucket yang menggunakan pengaturan object writer untuk Object Ownership	Grati	Man akse	N	-
% Kepemilikan Objek ember penulis objek	-	Persentase bucket yang menggunakan pengaturan object writer untuk Object Ownership	Grati	Man akse	Y	jumlah (ObjectOwnershipObjectWriterBucketCount) / jumlah () DistinctNumberOfBuckets
Transfer Akselerasi mengaktifkan jumlah bucket	TransferAccelerationEnabledBucketCount	Jumlah total bucket yang mengaktifkan Transfer Acceleration	Grati	Perf	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% Bucket yang diaktifkan Akselerasi Transfer	-	Persentase bucket yang mengaktifkan Transfer Acceleration	Grati	Perf	Y	jumlah (TransferAccelerationEnabledBucketCount) / jumlah () DistinctNumberOfBuckets
Pemberitahuan Acara mengaktifkan jumlah bucket	EventNotificationEnabledBucketCount	Jumlah total bucket yang mengaktifkan Pemberitahuan Acara	Grati	Keja	N	
% Pemberitahuan Acara mengaktifkan bucket	-	Persentase bucket yang mengaktifkan Pemberitahuan Acara	Grati	Keja	Y	jumlah (EventNotificationEnabledBucketCount) / jumlah () DistinctNumberOfBuckets
Semua permintaan	AllRequests	Jumlah total permintaan yang dibuat	Lanju	Aktif	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Dapatkan permintaan	GetRequests	Jumlah total `GET` permintaan yang dibuat	Lanju	Aktif	N	-
Masukkan permintaan	PutRequests	Jumlah total `PUT` permintaan yang dibuat	Lanju	Aktif	N	-
Permintaan kepala	HeadRequests	Jumlah total `HEAD` permintaan yang dibuat	Lanju	Aktif	N	-
Hapus permintaan	DeleteRequests	Jumlah total `DELETE` permintaan yang dibuat	Lanju	Aktif	N	-
Daftar permintaan	ListRequests	Jumlah total `LIST` permintaan yang dibuat	Lanju	Aktif	N	-
Permintaan posting	PostRequests	Jumlah total `POST` permintaan yang dibuat	Lanju	Aktif	N	-
Pilih permintaan	SelectRequests	Jumlah total permintaan S3 Select	Lanju	Aktif	N	-
Pilih byte yang dipindai	SelectScannedBytes	Jumlah byte S3 Pilih yang dipindai	Lanju	Aktif	N	-
Pilih byte yang dikembalikan	SelectReturnedBytes	Jumlah byte S3 Select dikembalikan	Lanju	Aktif	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
Byte yang diunduh	BytesDownloaded	Jumlah byte yang diunduh	Lanju	Aktif	N	-
% tingkat pengambilan	-	Persentase byte yang diunduh	Lanju	Aktif	Y	jumlah (BytesDownloaded) / jumlah () StorageBytes
Byte diunggah	BytesUploaded	Jumlah byte yang diunggah	Lanju	Aktif	N	-
% rasio konsumsi	-	Persentase byte yang diunggah	Lanju	Aktif	Y	jumlah (BytesUploaded) / jumlah () StorageBytes
Kesalahan 4xx	4xxKesalahan	Jumlah total kode status HTTP 4xx	Lanju	Aktif	N	-
Kesalahan 5xx	5xxKesalahan	Jumlah total kode status HTTP 5xx	Lanju	Aktif	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingl[1]	Kate[2]	Be	Form metri turur
Kesalahan total	-	Jumlah semua kesalahan 4xx dan 5xx	Lanju	Aktif	Y	jumlah (4xxErrors) + jumlah (5xxErrors)
% tingkat kesalahan	-	Jumlah total kesalahan 4xx dan 5xx sebagai persentase dari total permintaan	Lanju	Aktif	Y	jumlah (TotalErrors) / jumlah () TotalRequests
200 OK jumlah status	200OK StatusCount	Jumlah total 200 kode status OK	Lanju	Kode statu terpe i	N	-
% 200 status OK	-	Jumlah total 200 kode status OK sebagai persentase dari total permintaan	Lanju	Kode statu terpe i	Y	jumlah (200OKStatusCount) / jumlah () AllRequests
206 Jumlah status Konten Sebagian	206 PartialContentStatusCount	Jumlah total 206 kode status Konten Sebagian	Lanju	Kode statu terpe i	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% 206 Status Konten Sebagian	-	Jumlah total 206 kode status Konten Sebagian sebagai persentase dari total permintaan	Lanju	Kode statu terpe i	Y	jumlah (206PartialContentStatusCount) / jumlah () AllRequests
400 jumlah kesalahan Permintaan Buruk	400 BadRequestErrorCount	Jumlah total 400 kode status Permintaan Buruk	Lanju	Kode statu terpe i	N	-
% 400 Kesalahan Permintaan Buruk	-	Jumlah total 400 kode status Permintaan Buruk sebagai persentase dari total permintaan	Lanju	Kode statu terpe i	Y	jumlah (400BadRequestErrorCount) / jumlah () AllRequests
403 Jumlah kesalahan terlarang	403 Forbidden ErrorCount	Jumlah total 403 kode status Terlarang	Lanju	Kode statu terpe i	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingk[1]	Kate[2]	Be	Form metri turur
% 403 Kesalahan terlarang	-	Jumlah total 403 kode status Terlarang sebagai persentase dari total permintaan	Lanju	Kode statu terpe i	Y	jumlah (403ForbiddenErrorCount) / jumlah () AllRequests
404 Jumlah kesalahan Tidak Ditemukan	404 NotFoundErrorCount	Jumlah total 404 kode status Tidak Ditemukan	Lanju	Kode statu terpe i	N	-
% 404 Kesalahan Tidak Ditemukan	-	Jumlah total 404 kode status Tidak Ditemukan sebagai persentase dari total permintaan	Lanju	Kode statu terpe i	Y	jumlah (404NotFoundErrorCount) / jumlah () AllRequests
500 Jumlah Kesalahan Server Internal	500 InternalServerErrorCount	Jumlah total 500 kode status Kesalahan Server Internal	Lanju	Kode statu terpe i	N	-

Nama metrik	CloudWatch dan ekspor	Deskripsi	Tingl[1]	Kate[2]	Be	Form metri turur
% 500 Kesalahan Server Internal	-	Jumlah total 500 kode status Kesalahan Server Internal sebagai persentase dari total permintaan	Lanju	Kode statu terpe i	Y	jumlah (500InternalServerErrorCount) / jumlah () AllRequests
503 Layanan Jumlah kesalahan tidak tersedia	503 ServiceUnavailableErrorCount	Jumlah total 503 Kode status Layanan Tidak Tersedia	Lanju	Kode statu terpe i	N	-
% 503 Layanan Kesalahan tidak tersedia	-	Jumlah total 503 kode status Layanan Tidak Tersedia sebagai persentase dari total permintaan	Lanju	Kode statu terpe i	Y	jumlah (503ServiceUnavailableErrorCount) / jumlah () AllRequests

[1] Semua metrik penyimpanan tingkat gratis tersedia di tingkat grup Lensa Penyimpanan. Metrik tingkat lanjut tidak tersedia di tingkat grup Lensa Penyimpanan.

[2] Metrik hitungan aturan dan metrik pengaturan bucket tidak tersedia di tingkat awalan.

Bekerja dengan Amazon S3 Storage Lens dengan menggunakan konsol dan API

Amazon S3 Storage Lens adalah fitur analitik penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan objek. Anda dapat menggunakan metrik Lensa Penyimpanan S3 untuk menghasilkan wawasan ringkasan, seperti mencari tahu berapa banyak penyimpanan yang Anda miliki di seluruh organisasi atau bucket dan awalan mana yang paling cepat berkembang. Anda juga dapat menggunakan metrik Lensa Penyimpanan S3 untuk mengidentifikasi peluang pengoptimalan biaya, menerapkan praktik terbaik perlindungan data dan keamanan, serta meningkatkan kinerja beban kerja aplikasi. Misalnya, Anda dapat mengidentifikasi bucket yang tidak memiliki aturan Siklus Hidup S3 untuk mengakhiri unggahan multibagian yang tidak lengkap yang berusia lebih dari 7 hari. Anda juga dapat mengidentifikasi bucket yang tidak mengikuti praktik terbaik perlindungan data, seperti menggunakan Replikasi S3 atau Pembuatan Versi S3. S3 Storage Lens juga menganalisis metrik untuk memberikan rekomendasi kontekstual yang dapat Anda gunakan untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik untuk melindungi data Anda.

Lensa Penyimpanan S3 menggabungkan metrik Anda dan menampilkan informasi di bagian Snapshot akun di halaman Bucket konsol Amazon S3. S3 Storage Lens juga menyediakan dasbor interaktif yang dapat Anda gunakan untuk memvisualisasikan wawasan dan tren, menandai outlier, dan menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik perlindungan data. Dasbor Anda memiliki opsi penelusuran untuk menghasilkan dan memvisualisasikan wawasan di tingkat grup organisasi, akun, kelas penyimpanan, bucketWilayah AWS, awalan, atau Storage Lens. Anda juga dapat mengirim ekspor metrik harian dalam CSV atau Parquet format ke bucket S3.

Bagian berikut berisi contoh membuat, memperbarui, dan melihat konfigurasi S3 Storage Lens dan melakukan operasi yang terkait dengan fitur. Jika Anda menggunakan S3 Storage Lens dengan AWS Organizations, contoh-contoh ini juga mencakup kasus-kasus penggunaan tersebut. Dalam contoh, ganti nilai variabel dengan nilai yang khusus untuk Anda.

Topik

- Menggunakan Lensa Penyimpanan Amazon S3 di konsol
- Contoh Amazon S3 Storage Lens menggunakan AWS CLI
- Contoh Amazon S3 Storage Lens menggunakan .SDK for Java

Menggunakan Lensa Penyimpanan Amazon S3 di konsol

Lensa Penyimpanan Amazon S3 adalah fitur analisis penyimpanan cloud yang dapat Anda gunakan untuk mendapatkan visibilitas seluruh organisasi ke dalam penggunaan dan aktivitas penyimpanan objek. Anda dapat menggunakan metrik Lensa Penyimpanan S3 untuk menghasilkan wawasan ringkasan, seperti mencari tahu berapa banyak penyimpanan yang Anda miliki di seluruh organisasi atau bucket dan prefiks yang paling cepat berkembang. Anda juga dapat menggunakan metrik Lensa Penyimpanan S3 untuk mengidentifikasi peluang optimasi biaya, menerapkan praktik terbaik untuk perlindungan data dan keamanan, serta meningkatkan kinerja beban kerja aplikasi. Misalnya, Anda dapat mengidentifikasi bucket yang tidak memiliki aturan Siklus Hidup S3 untuk mengakhiri unggahan multibagian yang tidak lengkap dan berusia lebih dari 7 hari. Anda juga dapat mengidentifikasi bucket yang tidak mengikuti praktik terbaik perlindungan data, seperti menggunakan Replikasi S3 atau Penentuan Versi S3. Lensa Penyimpanan S3 menganalisis metrik untuk memberikan rekomendasi kontekstual yang dapat Anda gunakan untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik untuk melindungi data Anda.

Lensa Penyimpanan S3 menggabungkan metrik Anda dan menampilkan informasi di bagian Snapshot akun di halaman Bucket konsol Amazon S3. Lensa Penyimpanan S3 juga menyediakan dasbor interaktif yang dapat Anda gunakan untuk memvisualisasikan wawasan dan tren, menandai outlier, serta menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik perlindungan data. Dasbor Anda memiliki opsi perincian untuk menghasilkan dan memvisualisasikan wawasan di tingkat organisasi, akun, Wilayah AWS, kelas penyimpanan, bucket, prefiks, atau grup Lensa Penyimpanan. Anda juga dapat mengirimkan ekspor metrik harian dalam format CSV atau Parquet ke bucket S3.

> **Note**
>
> Setiap pembaruan konfigurasi dasbor dapat memakan waktu hingga 48 jam untuk ditampilkan atau divisualisasikan secara akurat.

Topik

- Membuat dan memperbarui dasbor Lensa Penyimpanan Amazon S3
- Menonaktifkan atau menghapus dasbor Lensa Penyimpanan Amazon S3
- Bekerja sama AWS Organizations untuk membuat dasbor tingkat organisasi

Membuat dan memperbarui dasbor Lensa Penyimpanan Amazon S3

Lensa Penyimpanan S3 menggabungkan metrik Anda dan menampilkan informasi di bagian Snapshot akun di halaman Bucket konsol Amazon S3. Lensa Penyimpanan S3 juga menyediakan dasbor interaktif yang dapat Anda gunakan untuk memvisualisasikan wawasan dan tren, menandai outlier, serta menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik perlindungan data. Dasbor Anda memiliki opsi perincian untuk menghasilkan dan memvisualisasikan wawasan di tingkat organisasi, akun, Wilayah AWS, kelas penyimpanan, bucket, prefiks, atau grup Lensa Penyimpanan. Anda juga dapat mengirimkan ekspor metrik harian dalam format CSV atau Parquet ke bucket S3.

Dasbor default Amazon S3 Storage Lens adalah. default-account-dashboard Dasbor ini telah dikonfigurasi sebelumnya oleh Amazon S3 untuk membantu memvisualisasikan ringkasan wawasan dan tren untuk gabungan metrik gratis dan lanjutan dari seluruh akun di konsol. Anda tidak dapat mengubah cakupan konfigurasi dasbor default, tapi Anda dapat meningkatkan pilihan metrik dari metrik gratis ke metrik dan rekomendasi tingkat lanjut berbayar, mengonfigurasi ekspor metrik opsional, atau bahkan menonaktifkan dasbor default. Dasbor default tidak dapat dihapus.

Anda juga dapat membuat dasbor kustom Lensa Penyimpanan S3 tambahan yang dapat dicakup ke organisasi Anda di AWS Organizations atau ke Wilayah atau bucket tertentu dalam akun.

Membuat dasbor Lensa Penyimpanan Amazon S3

Gunakan langkah-langkah berikut untuk membuat dasbor Lensa Penyimpanan Amazon S3 di konsol Amazon S3.

Langkah 1: Tentukan cakupan dasbor

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di bilah navigasi di bagian atas halaman, pilih nama AWS Wilayah yang saat ini ditampilkan. Selanjutnya, pilih Wilayah yang ingin Anda alihkan.
3. Di panel navigasi kiri, di bawah Lensa Penyimpanan S3, pilih Dasbor.
4. Pilih Buat dasbor.
5. Di laman Dasbor, di bagian Umum, lakukan hal-hal berikut:
    a. Lihat Wilayah Beranda untuk dasbor Anda. Wilayah beranda adalah Wilayah AWS tempat konfigurasi dan metrik untuk dasbor Lensa Penyimpanan ini disimpan.

b. Masukkan nama dasbor.

Nama dasbor harus kurang dari 65 karakter dan tidak boleh berisi karakter khusus atau spasi.

> **Note**
>
> Nama dasbor ini tidak dapat diubah setelah dasbor dibuat.

c. Secara opsional, Anda dapat memilih untuk menambahkan Tag ke dasbor. Anda dapat menggunakan tag untuk mengelola izin dasbor dan melacak biaya untuk Lensa Penyimpanan S3.

Untuk informasi lebih lanjut, lihat Mengontrol akses menggunakan tag sumber daya di Panduan Pengguna IAM dan AWS Tag Alokasi Biaya yang Dihasilkan di AWS Billing Panduan Pengguna.

> **Note**
>
> Anda dapat menambahkan hingga 50 tag ke konfigurasi dasbor.

6. Di bagian Cakupan dasbor, lakukan hal-hal berikut:

a. Pilih Wilayah dan bucket yang Anda inginkan untuk disertakan atau dikecualikan oleh Lensa Penyimpanan S3 di dasbor.

b. Pilih bucket di Wilayah terpilih yang Anda inginkan untuk disertakan atau dikecualikan oleh Lensa Penyimpanan S3. Anda dapat menyertakan atau mengecualikan bucket, tetapi tidak keduanya. Opsi ini tidak tersedia jika Anda membuat dasbor tingkat organisasi.

> **Note**
>
> - Anda dapat menyertakan atau mengecualikan Wilayah dan bucket. Pilihan ini terbatas pada Wilayah saja saat membuat dasbor tingkat organisasi di seluruh akun anggota di organisasi Anda.
> - Anda dapat memilih hingga 50 bucket untuk disertakan atau dikecualikan.

Langkah 2: Konfigurasikan pilihan metrik

1. Di bagian Pilihan metrik, pilih jenis metrik yang ingin digabungkan untuk dasbor ini.

   - Untuk menyertakan metrik gratis yang digabungkan di tingkat bucket dan tersedia untuk kueri selama 14 hari, pilih Metrik gratis.
   - Untuk dapat mengaktifkan metrik lanjutan dan opsi lanjutan lainnya, pilih Metrik dan rekomendasi lanjutan. Opsi ini mencakup agregasi awalan lanjutan, CloudWatch penerbitan Amazon, dan rekomendasi kontekstual. Data tersedia untuk kueri selama 15 bulan. Metrik dan rekomendasi lanjutan memiliki biaya tambahan. Untuk informasi selengkapnya, lihat Harga Amazon S3.

     Untuk informasi selengkapnya tentang metrik lanjutan dan metrik gratis, lihat Pilihan metrik.

2. Di bagian bawah Fitur metrik dan rekomendasi lanjutan, pilih opsi yang ingin diaktifkan:

   - Metrik lanjutan
   - CloudWatch penerbitan
   - Agregasi awalan

   > **Important**
   >
   > Jika Anda mengaktifkan agregasi awalan untuk konfigurasi Lensa Penyimpanan S3, metrik tingkat awalan tidak akan dipublikasikan. CloudWatch Hanya metrik Lensa Penyimpanan S3 tingkat bucket, akun, dan organisasi yang dipublikasikan. CloudWatch

3. Jika Anda mengaktifkan Metrik lanjutan, pilih Kategori metrik lanjutan yang ingin ditampilkan di dasbor Lensa Penyimpanan S3:

   - Metrik aktivitas
   - Metrik kode status terperinci
   - Metrik pengoptimalan biaya lanjutan
   - Metrik perlindungan data lanjutan

   Untuk informasi selengkapnya tentang kategori metrik, lihat Kategori metrik. Untuk daftar lengkap metrik, lihat Glosarium metrik Amazon S3 Storage Lens.

4. Jika Anda memilih untuk mengaktifkan agregasi awalan, konfigurasikan hal-hal berikut ini:

a. Pilih ukuran ambang batas awalan minimum untuk dasbor ini.

   Misalnya, ambang batas awalan sebesar 5 persen menunjukkan bahwa awalan yang membentuk 5 persen atau lebih dari total ukuran penyimpanan bucket akan digabungkan.

b. Pilih kedalaman awalan.

   Pengaturan ini menunjukkan jumlah maksimum level hingga awalan dapat dievaluasi. Kedalaman awalan harus kurang dari 10.

c. Masukkan karakter pembatas awalan.

   Nilai ini digunakan untuk mengidentifikasi setiap tingkat awalan. Nilai default di Amazon S3 adalah karakter `/`, tapi struktur penyimpanan Anda mungkin akan menggunakan karakter pembatas lain.

(Opsional) Langkah 3: Ekspor metrik untuk dasbor

1. Di bagian Ekspor metrik, untuk membuat ekspor metrik yang akan ditempatkan setiap hari di bucket tujuan pilihan Anda, pilih Aktifkan.

   Ekspor metrik berlaku dalam format CSV atau Apache Parquet. Ketentuan ini mewakili cakupan data yang sama dengan data dasbor Lensa Penyimpanan S3 Anda tanpa rekomendasi.

2. Jika ekspor metrik diaktifkan, pilih format output ekspor metrik harian Anda: CSV atau Apache Parquet.

   Parquet adalah format file sumber terbuka untuk Hadoop yang menyimpan data bertumpuk dalam format kolom datar.

3. Pilih bucket S3 tujuan untuk ekspor metrik Anda.

   Anda dapat memilih bucket di akun saat ini di dasbor Lensa Penyimpanan S3. Atau Anda dapat memilih yang lain Akun AWS jika Anda memiliki izin bucket tujuan dan ID akun pemilik bucket tujuan.

4. Pilih bucket S3 tujuan (format: `s3://bucket-name/prefix`).

   Bucket harus berada di Wilayah asal dasbor Lensa Penyimpanan S3 Anda. Konsol S3 menampilkan Izin bucket tujuan yang akan ditambahkan oleh Amazon S3 ke kebijakan bucket tujuan. Amazon S3 memperbarui kebijakan bucket di bucket tujuan agar S3 dapat menempatkan data di bucket tersebut.

5. (Opsional) Untuk mengaktifkan enkripsi di sisi server untuk ekspor metrik Anda, pilih Tentukan kunci enkripsi. Kemudian, pilih Jenis enkripsi: Kunci terkelola Amazon S3 (SSE-S3) atau AWS Key Management Service kunci (SSE-KMS).

   Anda dapat memilih antara kunci terkelola Amazon S3 (SSE-S3) dan kunci AWS Key Management Service (AWS KMS) (SSE-KMS).
6. (Opsional) Untuk menentukan AWS KMS kunci, Anda harus memilih kunci KMS atau memasukkan kunci Nama Sumber Daya Amazon (ARN).

   Jika Anda memilih CMK, Anda harus memberikan izin Lensa Penyimpanan S3 untuk dapat mengenkripsi di dalam kebijakan kunci AWS KMS . Untuk informasi selengkapnya, lihat Menggunakan AWS KMS key untuk mengenkripsi ekspor metrik Anda.
7. Pilih Buat dasbor.

Untuk mendapatkan visibilitas lebih lanjut ke penyimpanan, Anda dapat membuat satu atau beberapa grup Lensa Penyimpanan S3 dan melampirkannya ke dasbor. Grup Lensa Penyimpanan S3 adalah filter yang ditentukan khusus untuk objek berdasarkan awalan, akhiran, tag objek, ukuran objek, usia objek, atau kombinasi dari semua filter tersebut.

Anda dapat menggunakan grup Lensa Penyimpanan S3 untuk mendapatkan visibilitas terperinci ke dalam bucket bersama yang bersar, seperti danau data, untuk membuat keputusan bisnis yang lebih terinformasi. Misalnya, Anda dapat menyederhanakan alokasi penyimpanan dan mengoptimalkan pelaporan biaya dengan mengelompokkan penggunaan penyimpanan ke grup objek tertentu untuk masing-masing proyek dan pusat biaya dalam satu bucket atau beberapa bucket.

Untuk menggunakan grup Lensa Penyimpanan S3, Anda harus meningkatkan dasbor agar metrik dan rekomendasi tingkat lanjut dapat digunakan. Untuk informasi selengkapnya tentang grup Lensa Penyimpanan S3, lihat the section called “Bekerja dengan grup Lensa Penyimpanan”.

Memperbarui dasbor Lensa Penyimpanan Amazon S3

Gunakan langkah-langkah berikut untuk memperbarui dasbor Lensa Penyimpanan Amazon S3 di konsol Amazon S3.

Langkah 1: Perbarui cakupan dasbor

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi bagian kiri, pilih Lensa Penyimpanan, Dasbor.

3. Pilih dasbor yang ingin diedit, lalu pilih Edit.

   Laman Edit dasbor akan terbuka.

   > **Note**
   >
   > Anda tidak dapat mengubah hal-hal berikut:
   >
   > - Nama dasbor
   > - Wilayah asal
   > - Cakupan dasbor dari dasbor default, yang mencakup seluruh penyimpanan akun Anda

4. (Opsional) Di laman konfigurasi dasbor, di bagian Umum, perbarui dan tambahkan tag ke dasbor Anda.

   Anda dapat menggunakan tag untuk mengelola izin untuk dasbor Anda dan melacak biaya untuk Lensa Penyimpanan S3. Untuk informasi lebih lanjut, lihat Mengontrol akses menggunakan tag sumber daya di Panduan Pengguna IAM dan AWS Tag Alokasi Biaya yang Dihasilkan di AWS Billing Panduan Pengguna.

   > **Note**
   >
   > Anda dapat menambahkan hingga 50 tag ke konfigurasi dasbor.

5. Di bagian Cakupan dasbor, lakukan hal-hal berikut:

   a. Perbarui Wilayah dan bucket yang Anda inginkan untuk disertakan atau dikecualikan oleh Lensa Penyimpanan S3 di dasbor.

      > **Note**
      >
      > - Anda dapat menyertakan atau mengecualikan Wilayah dan bucket. Pilihan ini terbatas pada Wilayah saja saat membuat dasbor tingkat organisasi di seluruh akun anggota di organisasi Anda.
      > - Anda dapat memilih hingga 50 bucket untuk disertakan atau dikecualikan.

b. Perbarui bucket di Wilayah terpilih yang Anda inginkan untuk disertakan atau dikecualikan oleh Lensa Penyimpanan S3. Anda dapat menyertakan atau mengecualikan bucket, tetapi tidak keduanya. Opsi ini tidak tersedia saat dasbor tingkat organisasi dibuat.

Langkah 2: Perbarui pilihan metrik

1. Di bagian Pilihan metrik, pilih jenis metrik yang ingin digabungkan untuk dasbor ini.

   - Untuk menyertakan metrik gratis yang digabungkan di tingkat bucket dan tersedia untuk kueri selama 14 hari, pilih Metrik gratis.
   - Untuk dapat mengaktifkan metrik lanjutan dan opsi lanjutan lainnya, pilih Metrik dan rekomendasi lanjutan. Opsi ini mencakup agregasi awalan lanjutan, CloudWatch penerbitan Amazon, dan rekomendasi kontekstual. Data tersedia untuk kueri selama 15 bulan. Metrik dan rekomendasi lanjutan memiliki biaya tambahan. Untuk informasi selengkapnya, lihat Harga Amazon S3.

     Untuk informasi selengkapnya tentang metrik lanjutan dan metrik gratis, lihat Pilihan metrik.

2. Di bagian bawah Fitur metrik dan rekomendasi lanjutan, pilih opsi yang ingin diaktifkan:

   - Metrik lanjutan
   - CloudWatch penerbitan
   - Agregasi awalan

   > **Important**
   >
   > Jika Anda mengaktifkan agregasi awalan untuk konfigurasi Lensa Penyimpanan S3, metrik tingkat awalan tidak akan dipublikasikan. CloudWatch Hanya metrik Lensa Penyimpanan S3 tingkat bucket, akun, dan organisasi yang dipublikasikan. CloudWatch

3. Jika Anda mengaktifkan Metrik lanjutan, pilih Kategori metrik lanjutan yang ingin ditampilkan di dasbor Lensa Penyimpanan S3:

   - Metrik aktivitas
   - Metrik kode status terperinci
   - Metrik pengoptimalan biaya lanjutan
   - Metrik perlindungan data lanjutan

Untuk informasi selengkapnya tentang kategori metrik, lihat Kategori metrik. Untuk daftar lengkap metrik, lihat Glosarium metrik Amazon S3 Storage Lens.

4. Jika Anda memilih untuk mengaktifkan agregasi awalan, konfigurasikan hal-hal berikut ini:

   a. Pilih ukuran ambang batas awalan minimum untuk dasbor ini.

      Misalnya, ambang batas awalan sebesar 5 persen menunjukkan bahwa awalan yang membentuk 5 persen atau lebih dari total ukuran penyimpanan bucket akan digabungkan.

   b. Pilih kedalaman awalan.

      Pengaturan ini menunjukkan jumlah maksimum level hingga awalan dapat dievaluasi. Kedalaman awalan harus kurang dari 10.

   c. Masukkan karakter pembatas awalan.

      Ini adalah nilai yang digunakan untuk mengidentifikasi setiap tingkat awalan. Nilai default di Amazon S3 adalah karakter `/`, tapi struktur penyimpanan Anda mungkin akan menggunakan karakter pembatas lain.

(Opsional) Langkah 3: Ekspor metrik untuk dasbor

1. Di bagian Ekspor metrik, untuk membuat ekspor metrik yang akan ditempatkan setiap hari di bucket tujuan pilihan Anda, pilih Aktifkan. Untuk menonaktifkan ekspor metrik, pilih Nonaktifkan.

   Ekspor metrik berlaku dalam format CSV atau Apache Parquet. Ketentuan ini mewakili cakupan data yang sama dengan data dasbor Lensa Penyimpanan S3 Anda tanpa rekomendasi.

2. Jika sudah diaktifkan, pilih format output untuk ekspor metrik harian Anda: CSV atau Apache Parquet.

   Parquet adalah format file sumber terbuka untuk Hadoop yang menyimpan data bertumpuk dalam format kolom datar.

3. Pilih bucket S3 tujuan untuk ekspor metrik Anda.

   Anda dapat memilih bucket di akun saat ini di dasbor Lensa Penyimpanan S3. Atau Anda dapat memilih yang lain Akun AWS jika Anda memiliki izin bucket tujuan dan ID akun pemilik bucket tujuan.

4. Pilih bucket S3 tujuan (format: `s3://bucket-name/prefix`).

Bucket harus berada di Wilayah asal dasbor Lensa Penyimpanan S3 Anda. Konsol S3 menampilkan Izin bucket tujuan yang akan ditambahkan oleh Amazon S3 ke kebijakan bucket tujuan. Amazon S3 memperbarui kebijakan bucket di bucket tujuan agar S3 dapat menempatkan data di bucket tersebut.

5. (Opsional) Untuk mengaktifkan enkripsi di sisi server untuk ekspor metrik Anda, pilih Tentukan kunci enkripsi. Kemudian, pilih Jenis enkripsi: Kunci terkelola Amazon S3 (SSE-S3) atau AWS Key Management Service kunci (SSE-KMS).

   Anda dapat memilih antara kunci terkelola Amazon S3 (SSE-S3) dan kunci AWS Key Management Service (AWS KMS) (SSE-KMS).

6. (Opsional) Untuk menentukan AWS KMS kunci, Anda harus memilih kunci KMS atau memasukkan kunci Nama Sumber Daya Amazon (ARN). Di bawah AWS KMS kunci, tentukan kunci KMS Anda dengan salah satu cara berikut ini:

   - Untuk memilih dari daftar kunci KMS yang tersedia, pilih Pilih dari AWS KMS keys, dan pilih Kunci KMS Anda dari daftar kunci yang tersedia.

     Kunci Kunci yang dikelola AWS (`aws/s3`) dan kunci terkelola pelanggan Anda muncul dalam daftar ini. Untuk informasi selengkapnya tentang CMK, lihat Kunci pelanggan dan AWS kunci di AWS Key Management Service Panduan Pengembang.

     > **Note**
     >
     > The Kunci yang dikelola AWS (`aws/S3`) tidak didukung untuk enkripsi SSE-KMS dengan S3 Storage Lens.

   - Untuk memasukkan ARN kunci KMS, pilih Masukkan AWS KMS key ARN, dan masukkan ARN kunci KMS Anda di bidang yang muncul.
   - Untuk membuat kunci terkelola pelanggan baru di AWS KMS konsol, pilih Buat kunci KMS.

     Jika Anda memilih CMK, Anda harus memberikan izin Lensa Penyimpanan S3 untuk dapat mengenkripsi di dalam kebijakan kunci AWS KMS . Untuk informasi selengkapnya, lihat Menggunakan AWS KMS key untuk mengenkripsi ekspor metrik Anda.

     Untuk informasi selengkapnya tentang membuat AWS KMS key, lihat Membuat Kunci di Panduan AWS Key Management Service Pengembang.

7. Pilih Simpan perubahan.

Untuk mendapatkan visibilitas lebih lanjut ke penyimpanan, Anda dapat membuat satu atau beberapa grup Lensa Penyimpanan S3 dan melampirkannya ke dasbor. Grup Lensa Penyimpanan S3 adalah filter yang ditentukan khusus untuk objek berdasarkan awalan, akhiran, tag objek, ukuran objek, usia objek, atau kombinasi dari semua filter tersebut.

Anda dapat menggunakan grup Lensa Penyimpanan S3 untuk mendapatkan visibilitas terperinci ke dalam bucket bersama yang bersar, seperti danau data, untuk membuat keputusan bisnis yang lebih terinformasi. Misalnya, Anda dapat menyederhanakan alokasi penyimpanan dan mengoptimalkan pelaporan biaya dengan mengelompokkan penggunaan penyimpanan ke grup objek tertentu untuk masing-masing proyek dan pusat biaya dalam satu bucket atau beberapa bucket.

Untuk menggunakan grup Lensa Penyimpanan S3, Anda harus meningkatkan dasbor agar metrik dan rekomendasi tingkat lanjut dapat digunakan. Untuk informasi selengkapnya tentang grup Lensa Penyimpanan S3, lihat the section called “Bekerja dengan grup Lensa Penyimpanan”.

Menonaktifkan atau menghapus dasbor Lensa Penyimpanan Amazon S3

Lensa Penyimpanan S3 menggabungkan metrik Anda dan menampilkan informasi di bagian Snapshot akun di halaman Bucket konsol Amazon S3. Lensa Penyimpanan S3 juga menyediakan dasbor interaktif yang dapat Anda gunakan untuk memvisualisasikan wawasan dan tren, menandai outlier, serta menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik perlindungan data. Dasbor Anda memiliki opsi perincian untuk menghasilkan dan memvisualisasikan wawasan di tingkat organisasi, akun, Wilayah AWS, kelas penyimpanan, bucket, prefiks, atau grup Lensa Penyimpanan. Anda juga dapat mengirimkan ekspor metrik harian dalam format CSV atau Parquet ke bucket S3.

Dasbor default Amazon S3 Storage Lens adalah. default-account-dashboard Dasbor ini telah dikonfigurasi sebelumnya oleh Amazon S3 untuk membantu memvisualisasikan ringkasan wawasan dan tren untuk gabungan metrik gratis dan lanjutan dari seluruh akun di konsol. Anda tidak dapat mengubah cakupan konfigurasi dasbor default, tapi Anda dapat meningkatkan pilihan metrik dari metrik gratis ke metrik dan rekomendasi tingkat lanjut berbayar, mengonfigurasi ekspor metrik opsional, atau bahkan menonaktifkan dasbor default. Dasbor default tidak dapat dihapus.

Anda dapat menghapus atau menonaktifkan dasbor Lensa Penyimpanan Amazon S3 dari konsol Amazon S3. Menonaktifkan atau menghapus dasbor akan mencegah dasbor untuk menghasilkan metrik di masa mendatang. Dasbor yang dinonaktifkan masih dapat mempertahankan informasi konfigurasinya, sehingga dasbor dapat dilanjutkan dengan mudah ketika diaktifkan kembali. Dasbor yang dinonaktifkan akan mempertahankan data historisnya hingga tidak lagi tersedia untuk kueri.

Data untuk pilihan metrik gratis tersedia untuk kueri selama 14 hari, dan data untuk metrik lanjutan dan pilihan rekomendasi tersedia untuk kueri selama 15 bulan.

Menonaktifkan dasbor Lensa Penyimpanan Amazon S3

Untuk menonaktifkan dasbor Lensa Penyimpanan S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi bagian kiri, pilih Lensa Penyimpanan, Dasbor.
3. Di daftar Dasbor, pilih dasbor yang ingin dinonaktifkan, lalu pilih Nonaktifkan di bagian atas daftar.
4. Di laman konfirmasi, lakukan konfirmasi bahwa Anda ingin menonaktifkan dasbor dengan memasukkan nama dasbor ke dalam kolom teks, lalu pilih Konfirmasi.

Menghapus dasbor Lensa Penyimpanan Amazon S3

> **Note**
>
> Anda tidak dapat menghapus dasbor default. Namun, Anda dapat menonaktifkannya. Sebelum menghapus dasbor yang sudah dibuat, pertimbangkan hal-hal berikut:
>
> - Sebagai alternatif dari menghapus dasbor, Anda dapat menonaktifkan dasbor sehingga dasbor tersebut dapat tersedia untuk diaktifkan kembali di masa mendatang. Untuk informasi selengkapnya, lihat Menonaktifkan dasbor Lensa Penyimpanan Amazon S3.
> - Menghapus dasbor akan menghapus semua pengaturan konfigurasi yang terkait dengan dasbor.
> - Menghapus dasbor akan membuat semua data metrik historis tidak tersedia. Data historis ini masih dapat dipertahankan selama 15 bulan. Untuk mengakses data ini lagi, buat dasbor dengan nama yang sama di Wilayah asal yang sama dengan yang sudah dihapus.

Untuk menghapus dasbor Lensa Penyimpanan S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi bagian kiri, pilih Lensa Penyimpanan, Dasbor.

3. Di daftar Dasbor, pilih dasbor yang ingin dihapus, lalu pilih Hapus di bagian atas daftar.
4. Di laman Hapus dasbor, lakukan konfirmasi bahwa Anda ingin menghapus dasbor dengan memasukkan nama dasbor ke kolom teks. Lalu, pilih Konfirmasi.

Bekerja sama AWS Organizations untuk membuat dasbor tingkat organisasi

Lensa Penyimpanan S3 menggabungkan metrik Anda dan menampilkan informasi di bagian Snapshot akun di halaman Bucket konsol Amazon S3. Lensa Penyimpanan S3 juga menyediakan dasbor interaktif yang dapat Anda gunakan untuk memvisualisasikan wawasan dan tren, menandai outlier, serta menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik perlindungan data. Dasbor Anda memiliki opsi perincian untuk menghasilkan dan memvisualisasikan wawasan di tingkat organisasi, akun, Wilayah AWS, kelas penyimpanan, bucket, prefiks, atau grup Lensa Penyimpanan. Anda juga dapat mengirimkan ekspor metrik harian dalam format CSV atau Parquet ke bucket S3.

Dasbor default Amazon S3 Storage Lens adalah. default-account-dashboard Dasbor ini telah dikonfigurasi sebelumnya oleh Amazon S3 untuk membantu memvisualisasikan ringkasan wawasan dan tren untuk gabungan metrik gratis dan lanjutan dari seluruh akun di konsol. Anda tidak dapat mengubah cakupan konfigurasi dasbor default, tapi Anda dapat meningkatkan pilihan metrik dari metrik gratis ke metrik dan rekomendasi tingkat lanjut berbayar, mengonfigurasi ekspor metrik opsional, atau bahkan menonaktifkan dasbor default. Dasbor default tidak dapat dihapus.

Anda juga dapat membuat dasbor Lensa Penyimpanan S3 tambahan yang difokuskan pada bucket S3 tertentu Wilayah AWS, atau lainnya Akun AWS di organisasi Anda.

Dasbor Lensa Penyimpanan S3 menyediakan sumber daya informasi yang melimpah tentang cakupan penyimpanannya. Dasbor dapat memvisualisasikan lebih dari 30 metrik yang mewakili tren dan informasi, termasuk ringkasan penyimpanan, efisiensi biaya, perlindungan data, dan aktivitas.

Lensa Penyimpanan Amazon S3 dapat digunakan untuk mengumpulkan metrik penyimpanan dan data penggunaan untuk semua akun yang merupakan bagian dari hierarki Anda. AWS Organizations Untuk melakukan ini, Anda harus menggunakan AWS Organizations, dan Anda harus mengaktifkan akses tepercaya S3 Storage Lens dengan menggunakan akun AWS Organizations manajemen Anda.

Ketika akses tepercaya diaktifkan, Anda dapat menambahkan akses administrator yang didelegasikan ke akun di organisasi Anda. Akun ini kemudian dapat membuat dasbor dan konfigurasi di seluruh organisasi untuk Lensa Penyimpanan S3. Untuk informasi lebih lanjut tentang cara

mengaktifkan akses tepercaya, lihat Lensa Amazon S3 dan AWS Organizations di Panduan Pengguna AWS Organizations .

Kontrol konsol berikut hanya tersedia untuk akun AWS Organizations manajemen.

Mengaktifkan akses tepercaya untuk Lensa Penyimpanan S3 di organisasi Anda

Mengaktifkan akses tepercaya memungkinkan Amazon S3 Storage Lens mengakses hierarki, keanggotaan, dan struktur AWS Organizations AWS Organizations Anda melalui operasi API. Lensa Penyimpanan S3 adalah layanan tepercaya untuk seluruh struktur organisasi Anda. Layanan Penyimpanan S3 menciptakan peran terkait layanan di dalam manajemen organisasi atau akun administrator yang didelegasikan setiap kali konfigurasi dasbor dibuat.

Peran terkait layanan akan memberikan izin Lensa Penyimpanan S3 untuk menjelaskan organisasi, daftar akun, memverifikasi daftar akses layanan untuk organisasi, dan mendapatkan administrator yang didelegasikan untuk organisasi. Peran ini mengizinkan Lensa Penyimpanan S3 untuk mengumpulkan penggunaan penyimpanan lintas akun dan metrik aktivitas untuk dasbor di dalam akun di organisasi Anda.

Untuk informasi selengkapnya, lihat Menggunakan Peran Terkait Layanan untuk Lensa Penyimpanan Amazon S3.

> **Note**
>
> - Akses tepercaya hanya dapat diaktifkan oleh akun manajemen.
> - Hanya akun manajemen dan administrator yang didelegasikan yang dapat membuat dasbor atau konfigurasi Lensa Penyimpanan S3 untuk organisasi Anda.

Untuk mengaktifkan Lensa Penyimpanan S3 agar memiliki akses tepercaya

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi bagian kiri, pilih Lensa Penyimpanan, Pengaturan organisasi.
3. Di Akses organisasi, pilih Edit.

   Laman Akses organisasi akan terbuka. Di bagian ini, Anda dapat Mengaktifkan akses tepercaya untuk Lensa Penyimpanan S3. Fitur ini mengizinkan Anda dan setiap pemegang akun lain yang

ditambahkan sebagai administrator yang didelegasikan untuk dapat membuat dasbor untuk semua akun dan penyimpanan di organisasi Anda.

Menonaktifkan akses tepercaya Lensa Penyimpanan S3 di organisasi Anda

Menonaktifkan akses tepercaya akan membatasi Lensa Penyimpanan S3 untuk bekerja hanya pada tingkat akun. Setiap pemegang akun hanya dapat melihat manfaat Lensa Penyimpanan S3 terbatas pada cakupan akun mereka saja, dan bukan organisasi mereka. Setiap dasbor yang memerlukan akses tepercaya tidak akan diperbarui lagi, tetapi dasbor tersebut dapat mengkueri data historisnya masing-masing per periode saat data tersedia untuk kueri.

Menghapus akun sebagai administrator yang didelegasikan akan membatasi akses metrik dasbor Lensa Penyimpanan S3 untuk berfungsi hanya pada tingkat akun. Setiap dasbor organisasi yang dibuat tidak akan diperbarui lagi, tetapi dasbor tersebut dapat mengkueri data historisnya masing-masing per periode saat data tersedia untuk kueri.

> **Note**
>
> - Menonaktifkan akses tepercaya juga secara otomatis akan menonaktifkan semua dasbor tingkat organisasi karena Lensa Penyimpanan S3 tidak akan lagi memiliki akses tepercaya ke akun organisasi untuk mengumpulkan dan menggabungkan metrik penyimpanan.
> - Akun manajemen dan administrator yang didelegasikan masih dapat melihat data historis untuk dasbor yang dinonaktifkan ini dan dapat mengkueri data tersebut selagi tersedia.

Untuk menonaktifkan akses tepercaya untuk Lensa Penyimpanan S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi bagian kiri, pilih Lensa Penyimpanan, Pengaturan organisasi.
3. Di Akses organisasi, pilih Edit.

   Laman Akses organisasi akan terbuka. Di sini, Anda dapat Menonaktifkan akses tepercaya untuk Lensa Penyimpanan S3.

Mendaftarkan administrator yang didelegasikan untuk Lensa Penyimpanan S3

Setelah mengaktifkan akses tepercaya, Anda dapat mendaftarkan akses administrator delegasi ke akun di organisasi Anda. Ketika akun terdaftar sebagai administrator delegasi, akun akan menerima otorisasi untuk mengakses semua operasi API hanya-baca AWS Organizations . Hal ini memberikan visibilitas kepada anggota dan struktur organisasi Anda sehingga mereka dapat membuat dasbor Lensa Penyimpanan S3 atas nama Anda.

Untuk mendaftarkan administrator yang didelegasikan untuk Lensa Penyimpanan S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi bagian kiri, pilih Lensa Penyimpanan, Pengaturan organisasi.
3. Di bagian akses yang didelegasikan, untuk Akun, pilih Tambahkan akun.

   Laman Akses admin yang didelegasikan akan terbuka. Di sini, Anda dapat menambahkan ID Akun AWS sebagai administrator yang didelegasikan untuk membuat dasbor tingkat organisasi untuk semua akun dan penyimpanan di organisasi Anda.

Membatalkan pendaftaran administrator yang didelegasikan untuk Lensa Penyimpanan S3

Anda dapat membatalkan pendaftaran akses administrator delegasi ke akun di organisasi Anda. Ketika akun dideregistrasi sebagai administrator yang didelegasikan, akun kehilangan otorisasi untuk mengakses semua operasi AWS Organizations API hanya-baca yang memberikan visibilitas ke anggota dan struktur organisasi Anda.

> **Note**
>
> - Membatalkan pendaftaran administrator yang didelegasikan juga secara otomatis akan menonaktifkan semua dasbor tingkat organisasi yang dibuat oleh administrator yang didelegasikan.
> - Akun administrator delegasi masih dapat melihat data historis untuk dasbor yang dinonaktifkan ini sesuai dengan periode masing-masing data yang tersedia untuk kueri.

Untuk membatalkan pendaftaran akun untuk akses administrator yang didelegasikan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi bagian kiri, pilih Lensa Penyimpanan, Pengaturan organisasi.
3. Di bagian Akun dengan akses yang didelegasikan, pilih ID akun yang ingin batalkan pendaftarannya, lalu pilih Hapus.

Contoh Amazon S3 Storage Lens menggunakan AWS CLI

Lensa Penyimpanan S3 menggabungkan metrik Anda dan menampilkan informasi di bagian Snapshot akun di halaman Bucket konsol Amazon S3. S3 Storage Lens juga menyediakan dasbor interaktif yang dapat Anda gunakan untuk memvisualisasikan wawasan dan tren, menandai outlier, dan menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik perlindungan data. Dasbor Anda memiliki opsi penelusuran untuk menghasilkan dan memvisualisasikan wawasan di tingkat grup organisasi, akun, kelas penyimpanan, bucketWilayah AWS, awalan, atau Storage Lens. Anda juga dapat mengirim ekspor metrik harian dalam CSV atau Parquet format ke bucket S3. Untuk informasi selengkapnya, lihat Menilai aktivitas dan penggunaan penyimpanan dengan Amazon S3 Storage Lens.

Contoh berikut menunjukkan bagaimana Anda dapat menggunakan S3 Storage Lens dengan AWS Command Line Interface.

Topik

- File pembantu untuk menggunakan Amazon S3 Storage Lens
- Menggunakan konfigurasi Amazon S3 Storage Lens dengan AWS CLI
- Menggunakan Lensa Penyimpanan Amazon S3 dengan AWS Organizations contoh menggunakan AWS CLI

File pembantu untuk menggunakan Amazon S3 Storage Lens

Gunakan file JSON berikut dan input kuncinya untuk contoh Anda.

Contoh konfigurasi Lensa Penyimpanan S3 di JSON

Example **`config.json`**

`config.json`File tersebut berisi detail metrik lanjutan dan konfigurasi rekomendasi tingkat Organisasi Lensa Penyimpanan S3. Untuk menggunakan contoh berikut, ganti *`user input placeholders`* dengan informasi Anda sendiri.

> **Note**
>
> Biaya tambahan berlaku untuk metrik dan rekomendasi lanjutan. Untuk informasi selengkapnya, lihat metrik dan rekomendasi lanjutan.

```
{
  "Id": "SampleS3StorageLensConfiguration", //Use this property to identify your S3
 Storage Lens configuration.
  "AwsOrg": { //Use this property when enabling S3 Storage Lens for AWS Organizations.
    "Arn": "arn:aws:organizations::123456789012:organization/o-abcdefgh"
  },
  "AccountLevel": {
    "ActivityMetrics": {
      "IsEnabled":true
    },
    "AdvancedCostOptimizationMetrics": {
      "IsEnabled":true
    },
    "AdvancedDataProtectionMetrics": {
      "IsEnabled":true
    },
    "DetailedStatusCodesMetrics": {
      "IsEnabled":true
    },
    "BucketLevel": {
      "ActivityMetrics": {
        "IsEnabled":true
      },
      "AdvancedDataProtectionMetrics": {
      "IsEnabled":true
      },
      "AdvancedCostOptimizationMetrics": {
        "IsEnabled":true
```

```
      },
      "DetailedStatusCodesMetrics": {
        "IsEnabled":true
      },
      "PrefixLevel":{
        "StorageMetrics":{
          "IsEnabled":true,
          "SelectionCriteria":{
            "MaxDepth":5,
            "MinStorageBytesPercentage":1.25,
            "Delimiter":"/"
          }
        }
      }
    }
  },
  "Exclude": { //Replace with "Include" if you prefer to include Regions.
    "Regions": [
      "eu-west-1"
    ],
    "Buckets": [ //This attribute is not supported for AWS Organizations-level
 configurations.
      "arn:aws:s3:::source_bucket1"
    ]
  },
  "IsEnabled": true, //Whether the configuration is enabled
  "DataExport": { //Details about the metrics export
    "S3BucketDestination": {
      "OutputSchemaVersion": "V_1",
      "Format": "CSV", //You can add "Parquet" if you prefer.
      "AccountId": "111122223333",
      "Arn": "arn:aws:s3:::destination-bucket-name", // The destination bucket for your
 metrics export must be in the same Region as your S3 Storage Lens configuration.
      "Prefix": "prefix-for-your-export-destination",
      "Encryption": {
        "SSES3": {}
      }
    },
    "CloudWatchMetrics": {
      "IsEnabled": true
    }
  }
}
```

Contoh konfigurasi Lensa Penyimpanan S3 dengan grup Lensa Penyimpanan di JSON

Example **`config.json`**

`config.json`File berisi detail yang ingin Anda terapkan ke konfigurasi Lensa Penyimpanan Anda saat menggunakan grup Lensa Penyimpanan. Untuk menggunakan contoh, ganti *`user input placeholders`* dengan informasi Anda sendiri.

Untuk melampirkan semua grup Lensa Penyimpanan ke dasbor Anda, perbarui konfigurasi Lensa Penyimpanan Anda dengan sintaks berikut:

```
{
  "Id": "ExampleS3StorageLensConfiguration",
  "AccountLevel": {
    "ActivityMetrics": {
      "IsEnabled":true
    },
    "AdvancedCostOptimizationMetrics": {
      "IsEnabled":true
    },
    "AdvancedDataProtectionMetrics": {
      "IsEnabled":true
    },
    "BucketLevel": {
      "ActivityMetrics": {
      "IsEnabled":true
      },
    "StorageLensGroupLevel": {},
  "IsEnabled": true
}
```

Untuk menyertakan hanya dua grup Lensa Penyimpanan dalam konfigurasi dasbor Lensa Penyimpanan (*`slg-1 dan slg-2`*), gunakan sintaks berikut:

```
{
  "Id": "ExampleS3StorageLensConfiguration",
  "AccountLevel": {
    "ActivityMetrics": {
      "IsEnabled":true
    },
    "AdvancedCostOptimizationMetrics": {
      "IsEnabled":true
```

```
    },
    "AdvancedDataProtectionMetrics": {
      "IsEnabled":true
    },
    "BucketLevel": {
      "ActivityMetrics": {
      "IsEnabled":true
      },
   "StorageLensGroupLevel": {
        "SelectionCriteria": {
            "Include": [
                "arn:aws:s3:us-east-1:111122223333:storage-lens-group/slg-1",
                "arn:aws:s3:us-east-1:444455556666:storage-lens-group/slg-2"
            ]
    },
  "IsEnabled": true
}
```

Untuk mengecualikan hanya grup Lensa Penyimpanan tertentu agar tidak dilampirkan ke konfigurasi dasbor Anda, gunakan sintaks berikut:

```
{
  "Id": "ExampleS3StorageLensConfiguration",
  "AccountLevel": {
    "ActivityMetrics": {
      "IsEnabled":true
    },
    "AdvancedCostOptimizationMetrics": {
      "IsEnabled":true
    },
    "AdvancedDataProtectionMetrics": {
      "IsEnabled":true
    },
    "BucketLevel": {
      "ActivityMetrics": {
      "IsEnabled":true
      },
   "StorageLensGroupLevel": {
        "SelectionCriteria": {
            "Exclude": [
                "arn:aws:s3:us-east-1:111122223333:storage-lens-group/slg-1",
                "arn:aws:s3:us-east-1:444455556666:storage-lens-group/slg-2"
            ]
```

```
    },
  "IsEnabled": true
}
```

Contoh konfigurasi tag Lensa Penyimpanan S3 di JSON

Example **`tags.json`**

`tags.json`File berisi tag yang ingin Anda terapkan ke konfigurasi Lensa Penyimpanan S3 Anda. Untuk menggunakan contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
[
    {
        "Key": "key1",
        "Value": "value1"
    },
    {
        "Key": "key2",
        "Value": "value2"
    }
]
```

Contoh konfigurasi Lensa Penyimpanan S3 Izin IAM

Example **`permissions.json`**— Nama dasbor khusus

Kebijakan contoh ini menunjukkan `permissions.json` file IAM Lensa Penyimpanan S3 dengan nama dasbor tertentu yang ditentukan. Ganti *`value1us-east-1`*,*`your-dashboard-name`*,, dan *`example-account-id`* dengan nilai-nilai Anda sendiri.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetStorageLensConfiguration",
                "s3:DeleteStorageLensConfiguration",
                "s3:PutStorageLensConfiguration"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/key1": "value1"
```

```
                }
            },
            "Resource": "arn:aws:s3:us-east-1:example-account-id:storage-lens/your-
dashboard-name"
        }
    ]
}
```

Example **`permissions.json`**— Tidak ada nama dasbor khusus

Kebijakan contoh ini menunjukkan `permissions.json` file IAM Lensa Penyimpanan S3 tanpa nama dasbor tertentu yang ditentukan. Ganti *`value1us-east-1`*,, dan *`example-account-id`* dengan nilai-nilai Anda sendiri.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetStorageLensConfiguration",
                "s3:DeleteStorageLensConfiguration",
                "s3:PutStorageLensConfiguration"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/key1": "value1"
                }
            },
            "Resource": "arn:aws:s3:us-east-1:example-account-id:storage-lens/*"
        }
    ]
}
```

Menggunakan konfigurasi Amazon S3 Storage Lens dengan AWS CLI

Anda dapat menggunakan daftar AWS CLI untuk, membuat, menghapus, mendapatkan, menandai, dan memperbarui konfigurasi Lensa Penyimpanan S3 Anda. Contoh berikut menggunakan file JSON pembantu untuk input kunci. Untuk menggunakan contoh-contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

Buat konfigurasi Lensa Penyimpanan S3

Example Buat konfigurasi Lensa Penyimpanan S3

```
aws s3control put-storage-lens-configuration --account-id=111122223333 --
config-id=example-dashboard-configuration-id --region=us-east-1 --storage-lens-
configuration=file://./config.json --tags=file://./tags.json
```

Buat konfigurasi Lensa Penyimpanan S3 tanpa tag

Example Buat konfigurasi Lensa Penyimpanan S3 tanpa tag

```
aws s3control put-storage-lens-configuration --account-id=222222222222 --config-
id=your-configuration-id --region=us-east-1 --storage-lens-configuration=file://./
config.json
```

Dapatkan konfigurasi S3 Storage Lens

Example Dapatkan konfigurasi S3 Storage Lens

```
aws s3control get-storage-lens-configuration --account-id=222222222222 --config-
id=your-configuration-id --region=us-east-1
```

Buat daftar konfigurasi Lensa Penyimpanan S3 tanpa token berikutnya

Example Buat daftar konfigurasi Lensa Penyimpanan S3 tanpa token berikutnya

```
aws s3control list-storage-lens-configurations --account-id=222222222222 --region=us-
east-1
```

Cantumkan konfigurasi S3 Storage Lens

Example Cantumkan konfigurasi S3 Storage Lens

```
aws s3control list-storage-lens-configurations --account-id=222222222222 --region=us-
east-1 --next-token=abcdefghij1234
```

Menghapus konfigurasi S3 Storage Lens

Example Menghapus konfigurasi S3 Storage Lens

```
aws s3control delete-storage-lens-configuration --account-id=222222222222 --region=us-east-1 --config-id=your-configuration-id
```

Tambahkan tag ke konfigurasi Lensa Penyimpanan S3

Example Tambahkan tag ke konfigurasi Lensa Penyimpanan S3

```
aws s3control put-storage-lens-configuration-tagging --account-id=222222222222 --region=us-east-1 --config-id=your-configuration-id --tags=file://./tags.json
```

Dapatkan tag untuk konfigurasi S3 Storage Lens

Example Dapatkan tag untuk konfigurasi S3 Storage Lens

```
aws s3control get-storage-lens-configuration-tagging --account-id=222222222222 --region=us-east-1 --config-id=your-configuration-id
```

Menghapus tag untuk konfigurasi S3 Storage Lens

Example Menghapus tag untuk konfigurasi S3 Storage Lens

```
aws s3control delete-storage-lens-configuration-tagging --account-id=222222222222 --region=us-east-1 --config-id=your-configuration-id
```

Menggunakan Lensa Penyimpanan Amazon S3 dengan AWS Organizations contoh menggunakan AWS CLI

Gunakan Amazon S3 Storage Lens untuk mengumpulkan metrik penyimpanan dan data penggunaan untuk semua akun yang merupakan bagian dari hierarki AWS Organizations Anda. Untuk informasi lebih lanjut, lihat Menggunakan Amazon S3 Storage Lens dengan AWS Organizations.

Aktifkan akses tepercaya Organizations untuk S3 Storage Lens

Example Aktifkan akses tepercaya Organizations untuk S3 Storage Lens

```
aws organizations enable-aws-service-access --service-principal storage-lens.s3.amazonaws.com
```

Nonaktifkan akses tepercaya Organizations untuk S3 Storage Lens

Example Nonaktifkan akses tepercaya Organizations untuk S3 Storage Lens

```
aws organizations disable-aws-service-access --service-principal storage-
lens.s3.amazonaws.com
```

Daftarkan administrator yang didelegasikan Organizations untuk S3 Storage Lens

Example Daftarkan administrator yang didelegasikan Organizations untuk S3 Storage Lens

Untuk menggunakan contoh ini, ganti *111122223333* dengan Akun AWS ID yang sesuai.

```
aws organizations register-delegated-administrator --service-principal storage-
lens.s3.amazonaws.com --account-id 111122223333
```

Batalkan pendaftaran administrator yang didelegasikan Organizations untuk S3 Storage Lens

Example Batalkan pendaftaran administrator yang didelegasikan Organizations untuk S3 Storage Lens

Untuk menggunakan contoh ini, ganti *111122223333* dengan Akun AWS ID yang sesuai.

```
aws organizations deregister-delegated-administrator --service-principal storage-
lens.s3.amazonaws.com --account-id 111122223333
```

Contoh Amazon S3 Storage Lens menggunakan .SDK for Java

Lensa Penyimpanan S3 menggabungkan metrik Anda dan menampilkan informasi di bagian Snapshot akun di halaman Bucket konsol Amazon S3. S3 Storage Lens juga menyediakan dasbor interaktif yang dapat Anda gunakan untuk memvisualisasikan wawasan dan tren, menandai outlier, dan menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapkan praktik terbaik perlindungan data. Dasbor Anda memiliki opsi penelusuran untuk menghasilkan dan memvisualisasikan wawasan di tingkat grup organisasi, akun, kelas penyimpanan, bucketWilayah AWS, awalan, atau Storage Lens. Anda juga dapat mengirim ekspor metrik harian dalam CSV atau Parquet format ke bucket S3. Untuk informasi selengkapnya, lihat Menilai aktivitas dan penggunaan penyimpanan dengan Amazon S3 Storage Lens.

Contoh berikut menunjukkan bagaimana Anda dapat menggunakan S3 Storage Lens dengan AWS SDK for Java.

Topik

- Menggunakan konfigurasi Amazon S3 Storage Lens menggunakan SDK for Java

Menggunakan konfigurasi Amazon S3 Storage Lens menggunakan SDK for Java

Anda dapat menggunakan SDK for Java untuk mencantumkan, membuat, mendapatkan, dan memperbarui konfigurasi S3 Storage Lens Anda. Contoh berikut menggunakan file JSON pembantu untuk input kunci.

Topik

- Membuat dan memperbarui konfigurasi S3 Storage Lens
- Menghapus konfigurasi S3 Storage Lens
- Dapatkan konfigurasi S3 Storage Lens
- Cantumkan konfigurasi S3 Storage Lens
- Tambahkan tag ke konfigurasi Lensa Penyimpanan S3
- Dapatkan tag untuk konfigurasi S3 Storage Lens
- Menghapus tag untuk konfigurasi S3 Storage Lens
- Perbarui konfigurasi Lensa Penyimpanan S3 default dengan metrik dan rekomendasi lanjutan
- Pasang grup Lensa Penyimpanan ke dasbor Lensa Penyimpanan S3
- Menggunakan Amazon S3 Storage Lens dengan AWS Organizations contoh menggunakan SDK for Java

Membuat dan memperbarui konfigurasi S3 Storage Lens

Example Membuat dan memperbarui konfigurasi S3 Storage Lens

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.AccountLevel;
import com.amazonaws.services.s3control.model.ActivityMetrics;
import com.amazonaws.services.s3control.model.BucketLevel;
import com.amazonaws.services.s3control.model.CloudWatchMetrics;
```

```
import com.amazonaws.services.s3control.model.Format;
import com.amazonaws.services.s3control.model.Include;
import com.amazonaws.services.s3control.model.OutputSchemaVersion;
import com.amazonaws.services.s3control.model.PrefixLevel;
import com.amazonaws.services.s3control.model.PrefixLevelStorageMetrics;
import com.amazonaws.services.s3control.model.PutStorageLensConfigurationRequest;
import com.amazonaws.services.s3control.model.S3BucketDestination;
import com.amazonaws.services.s3control.model.SSES3;
import com.amazonaws.services.s3control.model.SelectionCriteria;
import com.amazonaws.services.s3control.model.StorageLensAwsOrg;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;
import com.amazonaws.services.s3control.model.StorageLensDataExport;
import com.amazonaws.services.s3control.model.StorageLensDataExportEncryption;
import com.amazonaws.services.s3control.model.StorageLensTag;

import java.util.Arrays;
import java.util.List;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class CreateAndUpdateDashboard {

    public static void main(String[] args) {
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";
        String exportAccountId = "Destination Account ID";
        String exportBucketArn = "arn:aws:s3:::destBucketName"; // The destination
 bucket for your metrics export must be in the same Region as your S3 Storage Lens
 configuration.
        String awsOrgARN = "arn:aws:organizations::123456789012:organization/o-
abcdefgh";
        Format exportFormat = Format.CSV;

        try {
            SelectionCriteria selectionCriteria = new SelectionCriteria()
                    .withDelimiter("/")
                    .withMaxDepth(5)
                    .withMinStorageBytesPercentage(10.0);
            PrefixLevelStorageMetrics prefixStorageMetrics = new
 PrefixLevelStorageMetrics()
                    .withIsEnabled(true)
                    .withSelectionCriteria(selectionCriteria);
            BucketLevel bucketLevel = new BucketLevel()
                    .withActivityMetrics(new ActivityMetrics().withIsEnabled(true))
```

```
                    .withAdvancedCostOptimizationMetrics(new
AdvancedCostOptimizationMetrics().withIsEnabled(true))
                    .withAdvancedDataProtectionMetrics(new
AdvancedDataProtectionMetrics().withIsEnabled(true))
                    .withDetailedStatusCodesMetrics(new
DetailedStatusCodesMetrics().withIsEnabled(true))
                    .withPrefixLevel(new
PrefixLevel().withStorageMetrics(prefixStorageMetrics));
            AccountLevel accountLevel = new AccountLevel()
                    .withActivityMetrics(new ActivityMetrics().withIsEnabled(true))
                    .withAdvancedCostOptimizationMetrics(new
AdvancedCostOptimizationMetrics().withIsEnabled(true))
                    .withAdvancedDataProtectionMetrics(new
AdvancedDataProtectionMetrics().withIsEnabled(true))
                    .withDetailedStatusCodesMetrics(new
DetailedStatusCodesMetrics().withIsEnabled(true))
                    .withBucketLevel(bucketLevel);

            Include include = new Include()
                    .withBuckets(Arrays.asList("arn:aws:s3:::bucketName"))
                    .withRegions(Arrays.asList("us-west-2"));

            StorageLensDataExportEncryption exportEncryption = new
StorageLensDataExportEncryption()
                    .withSSES3(new SSES3());
            S3BucketDestination s3BucketDestination = new S3BucketDestination()
                    .withAccountId(exportAccountId)
                    .withArn(exportBucketArn)
                    .withEncryption(exportEncryption)
                    .withFormat(exportFormat)
                    .withOutputSchemaVersion(OutputSchemaVersion.V_1)
                    .withPrefix("Prefix");
            CloudWatchMetrics cloudWatchMetrics = new CloudWatchMetrics()
                    .withIsEnabled(true);
            StorageLensDataExport dataExport = new StorageLensDataExport()
                    .withCloudWatchMetrics(cloudWatchMetrics)
                    .withS3BucketDestination(s3BucketDestination);

            StorageLensAwsOrg awsOrg = new StorageLensAwsOrg()
                    .withArn(awsOrgARN);

            StorageLensConfiguration configuration = new StorageLensConfiguration()
                    .withId(configurationId)
                    .withAccountLevel(accountLevel)
```

```
                    .withInclude(include)
                    .withDataExport(dataExport)
                    .withAwsOrg(awsOrg)
                    .withIsEnabled(true);

            List<StorageLensTag> tags = Arrays.asList(
                    new StorageLensTag().withKey("key-1").withValue("value-1"),
                    new StorageLensTag().withKey("key-2").withValue("value-2")
            );

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.putStorageLensConfiguration(new
 PutStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withStorageLensConfiguration(configuration)
                    .withTags(tags)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Menghapus konfigurasi S3 Storage Lens

Example Menghapus konfigurasi S3 Storage Lens

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
```

```
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.DeleteStorageLensConfigurationRequest;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class DeleteDashboard {

    public static void main(String[] args) {
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";
        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.deleteStorageLensConfiguration(new
 DeleteStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Dapatkan konfigurasi S3 Storage Lens

Example Dapatkan konfigurasi S3 Storage Lens

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
```

```
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.GetStorageLensConfigurationRequest;
import com.amazonaws.services.s3control.model.GetStorageLensConfigurationResult;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class GetDashboard {

    public static void main(String[] args) {
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";

        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            final StorageLensConfiguration configuration =
                    s3ControlClient.getStorageLensConfiguration(new
 GetStorageLensConfigurationRequest()
                            .withAccountId(sourceAccountId)
                            .withConfigId(configurationId)
                    ).getStorageLensConfiguration();

            System.out.println(configuration.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Cantumkan konfigurasi S3 Storage Lens

Example Cantumkan konfigurasi S3 Storage Lens

```
package aws.example.s3control;
```

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.ListStorageLensConfigurationEntry;
import com.amazonaws.services.s3control.model.ListStorageLensConfigurationsRequest;

import java.util.List;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class ListDashboard {

    public static void main(String[] args) {
        String sourceAccountId = "Source Account ID";
        String nextToken = "nextToken";

        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            final List<ListStorageLensConfigurationEntry> configurations =
                    s3ControlClient.listStorageLensConfigurations(new
 ListStorageLensConfigurationsRequest()
                            .withAccountId(sourceAccountId)
                            .withNextToken(nextToken)
                    ).getStorageLensConfigurationList();

            System.out.println(configurations.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
```

```
}
```

Tambahkan tag ke konfigurasi Lensa Penyimpanan S3

Example Tambahkan tag ke konfigurasi Lensa Penyimpanan S3

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import
 com.amazonaws.services.s3control.model.PutStorageLensConfigurationTaggingRequest;
import com.amazonaws.services.s3control.model.StorageLensTag;

import java.util.Arrays;
import java.util.List;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class PutDashboardTagging {

    public static void main(String[] args) {
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";

        try {
            List<StorageLensTag> tags = Arrays.asList(
                    new StorageLensTag().withKey("key-1").withValue("value-1"),
                    new StorageLensTag().withKey("key-2").withValue("value-2")
            );

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.putStorageLensConfigurationTagging(new
 PutStorageLensConfigurationTaggingRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withTags(tags)
```

```
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Dapatkan tag untuk konfigurasi S3 Storage Lens

Example Dapatkan tag untuk konfigurasi S3 Storage Lens

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.DeleteStorageLensConfigurationRequest;
import
 com.amazonaws.services.s3control.model.GetStorageLensConfigurationTaggingRequest;
import com.amazonaws.services.s3control.model.StorageLensTag;

import java.util.List;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class GetDashboardTagging {

    public static void main(String[] args) {
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";
        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();
```

```
            final List<StorageLensTag> s3Tags = s3ControlClient
                    .getStorageLensConfigurationTagging(new
 GetStorageLensConfigurationTaggingRequest()
                            .withAccountId(sourceAccountId)
                            .withConfigId(configurationId)
                    ).getTags();

            System.out.println(s3Tags.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Menghapus tag untuk konfigurasi S3 Storage Lens

Example Menghapus tag untuk konfigurasi S3 Storage Lens

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import
 com.amazonaws.services.s3control.model.DeleteStorageLensConfigurationTaggingRequest;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class DeleteDashboardTagging {

    public static void main(String[] args) {
        String configurationId = "ConfigurationId";
        String sourceAccountId = "Source Account ID";
        try {
            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
```

```
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.deleteStorageLensConfigurationTagging(new
 DeleteStorageLensConfigurationTaggingRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Perbarui konfigurasi Lensa Penyimpanan S3 default dengan metrik dan rekomendasi lanjutan

Example Perbarui konfigurasi Lensa Penyimpanan S3 default dengan metrik dan rekomendasi lanjutan

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.AccountLevel;
import com.amazonaws.services.s3control.model.ActivityMetrics;
import com.amazonaws.services.s3control.model.BucketLevel;
import com.amazonaws.services.s3control.model.Format;
import com.amazonaws.services.s3control.model.Include;
import com.amazonaws.services.s3control.model.OutputSchemaVersion;
import com.amazonaws.services.s3control.model.PrefixLevel;
import com.amazonaws.services.s3control.model.PrefixLevelStorageMetrics;
import com.amazonaws.services.s3control.model.PutStorageLensConfigurationRequest;
import com.amazonaws.services.s3control.model.S3BucketDestination;
import com.amazonaws.services.s3control.model.SSES3;
```

```
import com.amazonaws.services.s3control.model.SelectionCriteria;
import com.amazonaws.services.s3control.model.StorageLensAwsOrg;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;
import com.amazonaws.services.s3control.model.StorageLensDataExport;
import com.amazonaws.services.s3control.model.StorageLensDataExportEncryption;
import com.amazonaws.services.s3control.model.StorageLensTag;

import java.util.Arrays;
import java.util.List;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class UpdateDefaultConfigWithPaidFeatures {

    public static void main(String[] args) {
        String configurationId = "default-account-dashboard"; // This configuration ID
 cannot be modified.
        String sourceAccountId = "Source Account ID";

        try {
            SelectionCriteria selectionCriteria = new SelectionCriteria()
                    .withDelimiter("/")
                    .withMaxDepth(5)
                    .withMinStorageBytesPercentage(10.0);
            PrefixLevelStorageMetrics prefixStorageMetrics = new
 PrefixLevelStorageMetrics()
                    .withIsEnabled(true)
                    .withSelectionCriteria(selectionCriteria);
            BucketLevel bucketLevel = new BucketLevel()
                    .withActivityMetrics(new ActivityMetrics().withIsEnabled(true))
                    .withPrefixLevel(new
 PrefixLevel().withStorageMetrics(prefixStorageMetrics));
            AccountLevel accountLevel = new AccountLevel()
                    .withActivityMetrics(new ActivityMetrics().withIsEnabled(true))
                    .withBucketLevel(bucketLevel);

            StorageLensConfiguration configuration = new StorageLensConfiguration()
                    .withId(configurationId)
                    .withAccountLevel(accountLevel)
                    .withIsEnabled(true);

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
```

```
                    .build();

            s3ControlClient.putStorageLensConfiguration(new
 PutStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withStorageLensConfiguration(configuration)
            );

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

> **Note**
>
> Biaya tambahan berlaku untuk metrik dan rekomendasi lanjutan. Untuk informasi selengkapnya, lihat metrik dan rekomendasi lanjutan.

Pasang grup Lensa Penyimpanan ke dasbor Lensa Penyimpanan S3

Example Pasang semua grup Lensa Penyimpanan ke dasbor

Contoh SDK for Java berikut ini melampirkan semua grup Storage Lens di *akun* 111122223333 ke dasbor: *DashBoardConfigurationId*

```
package aws.example.s3control;


import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.BucketLevel;
```

```
import com.amazonaws.services.s3control.model.PutStorageLensConfigurationRequest;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.model.AccountLevel;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;
import com.amazonaws.services.s3control.model.StorageLensGroupLevel;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class CreateDashboardWithStorageLensGroups {
    public static void main(String[] args) {
        String configurationId = "ExampleDashboardConfigurationId";
        String sourceAccountId = "111122223333";

        try {
            StorageLensGroupLevel storageLensGroupLevel = new StorageLensGroupLevel();

            AccountLevel accountLevel = new AccountLevel()
                    .withBucketLevel(new BucketLevel())
                    .withStorageLensGroupLevel(storageLensGroupLevel);

            StorageLensConfiguration configuration = new StorageLensConfiguration()
                    .withId(configurationId)
                    .withAccountLevel(accountLevel)
                    .withIsEnabled(true);

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.putStorageLensConfiguration(new
 PutStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withStorageLensConfiguration(configuration)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
```

```
        }
    }
}
```

Example Pasang dua grup Lensa Penyimpanan ke dasbor

AWS SDK for JavaContoh berikut melampirkan dua grup Storage Lens (*StorageLensGroupName1* dan *StorageLensGroupName2*) ke *ExampleDashboardConfigurationId*dasbor.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.AccountLevel;
import com.amazonaws.services.s3control.model.BucketLevel;
import com.amazonaws.services.s3control.model.PutStorageLensConfigurationRequest;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;
import com.amazonaws.services.s3control.model.StorageLensGroupLevel;
import com.amazonaws.services.s3control.model.StorageLensGroupLevelSelectionCriteria;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class CreateDashboardWith2StorageLensGroups {
    public static void main(String[] args) {
        String configurationId = "ExampleDashboardConfigurationId";
        String storageLensGroupName1 = "StorageLensGroupName1";
        String storageLensGroupName2 = "StorageLensGroupName2";
        String sourceAccountId = "111122223333";

        try {
            StorageLensGroupLevelSelectionCriteria selectionCriteria = new
 StorageLensGroupLevelSelectionCriteria()
                    .withInclude(
                            "arn:aws:s3:" + US_WEST_2.getName() + ":" + sourceAccountId
 + ":storage-lens-group/" + storageLensGroupName1,
                            "arn:aws:s3:" + US_WEST_2.getName() + ":" + sourceAccountId
 + ":storage-lens-group/" + storageLensGroupName2);

            System.out.println(selectionCriteria);
            StorageLensGroupLevel storageLensGroupLevel = new StorageLensGroupLevel()
```

```
                    .withSelectionCriteria(selectionCriteria);

            AccountLevel accountLevel = new AccountLevel()
                    .withBucketLevel(new BucketLevel())
                    .withStorageLensGroupLevel(storageLensGroupLevel);

            StorageLensConfiguration configuration = new StorageLensConfiguration()
                    .withId(configurationId)
                    .withAccountLevel(accountLevel)
                    .withIsEnabled(true);

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.putStorageLensConfiguration(new
 PutStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withStorageLensConfiguration(configuration)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example Lampirkan semua grup Lensa Penyimpanan dengan pengecualian

Contoh SDK for Java berikut ini melampirkan semua grup Storage Lens ke ExampleDashboardConfigurationIddasbor, tidak termasuk dua yang ditentukan StorageLensGroupName(1 StorageLensGroupName dan 2):

```
package aws.example.s3control;
```

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;
import com.amazonaws.services.s3control.model.AccountLevel;
import com.amazonaws.services.s3control.model.BucketLevel;
import com.amazonaws.services.s3control.model.PutStorageLensConfigurationRequest;
import com.amazonaws.services.s3control.model.StorageLensConfiguration;
import com.amazonaws.services.s3control.model.StorageLensGroupLevel;
import com.amazonaws.services.s3control.model.StorageLensGroupLevelSelectionCriteria;

import static com.amazonaws.regions.Regions.US_WEST_2;

public class CreateDashboardWith2StorageLensGroupsExcluded {
    public static void main(String[] args) {
        String configurationId = "ExampleDashboardConfigurationId";
        String storageLensGroupName1 = "StorageLensGroupName1";
        String storageLensGroupName2 = "StorageLensGroupName2";
        String sourceAccountId = "111122223333";

        try {
            StorageLensGroupLevelSelectionCriteria selectionCriteria = new
 StorageLensGroupLevelSelectionCriteria()
                    .withInclude(
                            "arn:aws:s3:" + US_WEST_2.getName() + ":" + sourceAccountId
 + ":storage-lens-group/" + storageLensGroupName1,
                            "arn:aws:s3:" + US_WEST_2.getName() + ":" + sourceAccountId
 + ":storage-lens-group/" + storageLensGroupName2);

            System.out.println(selectionCriteria);
            StorageLensGroupLevel storageLensGroupLevel = new StorageLensGroupLevel()
                    .withSelectionCriteria(selectionCriteria);

            AccountLevel accountLevel = new AccountLevel()
                    .withBucketLevel(new BucketLevel())
                    .withStorageLensGroupLevel(storageLensGroupLevel);

            StorageLensConfiguration configuration = new StorageLensConfiguration()
                    .withId(configurationId)
                    .withAccountLevel(accountLevel)
                    .withIsEnabled(true);

            AWSS3Control s3ControlClient = AWSS3ControlClient.builder()
```

```
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(US_WEST_2)
                    .build();

            s3ControlClient.putStorageLensConfiguration(new
 PutStorageLensConfigurationRequest()
                    .withAccountId(sourceAccountId)
                    .withConfigId(configurationId)
                    .withStorageLensConfiguration(configuration)
            );
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Menggunakan Amazon S3 Storage Lens dengan AWS Organizations contoh menggunakan SDK for Java

Gunakan Amazon S3 Storage Lens untuk mengumpulkan metrik penyimpanan dan data penggunaan untuk semua akun yang merupakan bagian dari hierarki AWS Organizations Anda. Untuk informasi lebih lanjut, lihat Menggunakan Amazon S3 Storage Lens dengan AWS Organizations.

Topik

- Aktifkan akses tepercaya Organizations untuk S3 Storage Lens
- Nonaktifkan akses tepercaya Organizations untuk S3 Storage Lens
- Daftarkan administrator yang didelegasikan Organizations untuk S3 Storage Lens
- Batalkan pendaftaran administrator yang didelegasikan Organizations untuk S3 Storage Lens

Aktifkan akses tepercaya Organizations untuk S3 Storage Lens

Example Aktifkan akses tepercaya Organizations untuk S3 Storage Lens

```
import com.amazonaws.AmazonServiceException;
```

```
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.organizations.AWSOrganizations;
import com.amazonaws.services.organizations.AWSOrganizationsClient;
import com.amazonaws.services.organizations.model.EnableAWSServiceAccessRequest;

public class EnableOrganizationsTrustedAccess {
 private static final String S3_STORAGE_LENS_SERVICE_PRINCIPAL = "storage-
lens.s3.amazonaws.com";

 public static void main(String[] args) {
  try {
            AWSOrganizations organizationsClient = AWSOrganizationsClient.builder()
                .withCredentials(new ProfileCredentialsProvider())
                .withRegion(Regions.US_EAST_1)
                .build();

            organizationsClient.enableAWSServiceAccess(new
 EnableAWSServiceAccessRequest()
                .withServicePrincipal(S3_STORAGE_LENS_SERVICE_PRINCIPAL));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but AWS Organizations couldn't
 process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // AWS Organizations couldn't be contacted for a response, or the client
            // couldn't parse the response from AWS Organizations.
            e.printStackTrace();
        }
 }
}
```

Nonaktifkan akses tepercaya Organizations untuk S3 Storage Lens

Example Nonaktifkan akses tepercaya Organizations untuk S3 Storage Lens

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.organizations.AWSOrganizations;
import com.amazonaws.services.organizations.AWSOrganizationsClient;
```

```
import com.amazonaws.services.organizations.model.DisableAWSServiceAccessRequest;

public class DisableOrganizationsTrustedAccess {
 private static final String S3_STORAGE_LENS_SERVICE_PRINCIPAL = "storage-
lens.s3.amazonaws.com";

 public static void main(String[] args) {
  try {
            AWSOrganizations organizationsClient = AWSOrganizationsClient.builder()
                .withCredentials(new ProfileCredentialsProvider())
                .withRegion(Regions.US_EAST_1)
                .build();

            // Make sure to remove any existing delegated administrator for S3 Storage
 Lens
            // before disabling access; otherwise, the request will fail.
            organizationsClient.disableAWSServiceAccess(new
 DisableAWSServiceAccessRequest()
                .withServicePrincipal(S3_STORAGE_LENS_SERVICE_PRINCIPAL));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but AWS Organizations couldn't
 process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // AWS Organizations couldn't be contacted for a response, or the client
            // couldn't parse the response from AWS Organizations.
            e.printStackTrace();
        }
 }
}
```

Daftarkan administrator yang didelegasikan Organizations untuk S3 Storage Lens

Example Daftarkan administrator yang didelegasikan Organizations untuk S3 Storage Lens

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.organizations.AWSOrganizations;
import com.amazonaws.services.organizations.AWSOrganizationsClient;
import
 com.amazonaws.services.organizations.model.RegisterDelegatedAdministratorRequest;
```

```
public class RegisterOrganizationsDelegatedAdministrator {
 private static final String S3_STORAGE_LENS_SERVICE_PRINCIPAL = "storage-
lens.s3.amazonaws.com";

 public static void main(String[] args) {
  try {
            String delegatedAdminAccountId = "111122223333"; // Account Id for the
 delegated administrator.
            AWSOrganizations organizationsClient = AWSOrganizationsClient.builder()
                .withCredentials(new ProfileCredentialsProvider())
                .withRegion(Regions.US_EAST_1)
                .build();

            organizationsClient.registerDelegatedAdministrator(new
 RegisterDelegatedAdministratorRequest()
                .withAccountId(delegatedAdminAccountId)
                .withServicePrincipal(S3_STORAGE_LENS_SERVICE_PRINCIPAL));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but AWS Organizations couldn't
 process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // AWS Organizations couldn't be contacted for a response, or the client
            // couldn't parse the response from AWS Organizations.
            e.printStackTrace();
        }
 }
}
```

Batalkan pendaftaran administrator yang didelegasikan Organizations untuk S3 Storage Lens

Example Batalkan pendaftaran administrator yang didelegasikan Organizations untuk S3 Storage Lens

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.organizations.AWSOrganizations;
import com.amazonaws.services.organizations.AWSOrganizationsClient;
```

```
import
 com.amazonaws.services.organizations.model.DeregisterDelegatedAdministratorRequest;

public class DeregisterOrganizationsDelegatedAdministrator {
 private static final String S3_STORAGE_LENS_SERVICE_PRINCIPAL = "storage-
lens.s3.amazonaws.com";

 public static void main(String[] args) {
  try {
            String delegatedAdminAccountId = "111122223333"; // Account Id for the
 delegated administrator.
            AWSOrganizations organizationsClient = AWSOrganizationsClient.builder()
                .withCredentials(new ProfileCredentialsProvider())
                .withRegion(Regions.US_EAST_1)
                .build();

            organizationsClient.deregisterDelegatedAdministrator(new
 DeregisterDelegatedAdministratorRequest()
                .withAccountId(delegatedAdminAccountId)
                .withServicePrincipal(S3_STORAGE_LENS_SERVICE_PRINCIPAL));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but AWS Organizations couldn't
 process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // AWS Organizations couldn't be contacted for a response, or the client
            // couldn't parse the response from AWS Organizations.
            e.printStackTrace();
        }
 }
}
```

Bekerja dengan grup Lensa Penyimpanan

Grup Lensa Penyimpanan Amazon S3 menggabungkan metrik dengan menggunakan filter khusus berdasarkan metadata objek. Grup Lensa Penyimpanan membantu menelusuri karakteristik data, seperti distribusi objek berdasarkan usia, jenis file yang paling umum, dan banyak lagi. Misalnya, Anda dapat memfilter metrik berdasarkan tag objek untuk mengidentifikasi set data yang tumbuh paling cepat, atau memvisualisasikan penyimpanan berdasarkan ukuran dan usia objek untuk menginformasikan strategi arsip penyimpanan. Dengan begitu, grup Lensa Penyimpanan Amazon S3 membantu Anda untuk lebih memahami dan mengoptimalkan penyimpanan S3.

Saat menggunakan grup Lensa Penyimpanan, Anda dapat menganalisis dan memfilter metrik Lensa Penyimpanan S3 dengan menggunakan metadata objek seperti awalan, akhiran, tag objek, ukuran objek, atau usia objek. Anda juga dapat menerapkan kombinasi filter tersebut. Setelah melampirkan grup Lensa Penyimpanan ke dasbor Lensa Penyimpanan S3, Anda dapat melihat metrik Lensa Penyimpanan S3 yang dikumpulkan oleh grup Lensa Penyimpanan Amazon S3 secara langsung di dasbor.

Misalnya, Anda juga dapat memfilter metrik berdasarkan ukuran objek atau kelompok usia untuk menentukan bagian penyimpanan mana yang terdiri dari objek kecil. Lalu, Anda dapat menggunakan informasi ini dengan S3 Intelligent-Tiering atau S3 Lifecycle untuk mentransisikan objek kecil ke kelas penyimpanan yang berbeda untuk pengoptimalan biaya dan penyimpanan.

Topik

- Cara kerja grup Lensa Penyimpanan S3
- Menggunakan grup Lensa Penyimpanan

Cara kerja grup Lensa Penyimpanan S3

Anda dapat menggunakan grup Lensa Penyimpanan untuk menggabungkan metrik dengan menggunakan filter khusus berdasarkan metadata objek. Saat Anda menentukan filter kustom, Anda dapat menggunakan awalan, akhiran, tag objek, ukuran objek, usia objek, atau kombinasi dari filter kustom tersebut. Selama pembuatan grup Lensa Penyimpanan, Anda juga dapat menyertakan satu atau beberapa syarat filter. Untuk menentukan beberapa syarat filter, Anda dapat menggunakan `And` atau operator logis `Or`.

Saat Anda membuat dan mengonfigurasi grup Lensa Penyimpanan, grup Lensa Penyimpanan itu sendiri bertindak sebagai filter kustom di dasbor tempat grup dilampirkan. Di dasbor, Anda kemudian dapat menggunakan filter grup Lensa Penyimpanan untuk mendapatkan metrik penyimpanan berdasarkan filter kustom yang ditentukan di dalam grup.

Untuk melihat data untuk grup Lensa Penyimpanan di dasbor Lensa Penyimpanan S3, Anda harus melampirkan grup ke dasbor setelah grup tersebut dibuat. Setelah grup Lensa Penyimpanan terpasang ke dasbor Lensa Penyimpanan, dasbor akan mengumpulkan metrik penggunaan penyimpanan dalam waktu 48 jam. Anda kemudian dapat memvisualisasikan data tersebut di dasbor Lensa Penyimpanan atau mengekspornya melalui ekspor metrik. Jika Anda lupa melampirkan grup Lensa Penyimpanan ke dasbor, data grup Lensa Penyimpanan Anda tidak akan diambil atau ditampilkan di mana pun.

> **Note**
>
> - Saat Anda membuat grup Lensa Penyimpanan S3, Anda membuat sumber daya AWS. Oleh karena itu, setiap grup Lensa Penyimpanan memiliki Amazon Resource Name (ARN) sendiri, yang dapat ditentukan saat melampirkannya atau mengecualikannya dari dasbor Lensa Penyimpanan S3.
> - Jika grup Lensa Penyimpanan tidak dilampirkan ke dasbor, Anda tidak akan dikenakan biaya tambahan untuk membuat grup Lensa Penyimpanan.
> - Lensa Penyimpanan S3 menggabungkan metrik penggunaan untuk objek di bawah semua grup Lensa Penyimpanan yang cocok. Oleh karena itu, jika suatu objek cocok dengan syarat filter untuk dua atau lebih grup Lensa Penyimpanan, Anda akan melihat penghitungan berulang untuk objek yang sama di seluruh penggunaan penyimpanan.

Anda dapat membuat grup Lensa Penyimpanan di tingkat akun di Wilayah asal tertentu (dari daftar Wilayah AWS yang didukung). Lalu, Anda dapat melampirkan grup Lensa Penyimpanan ke beberapa dasbor Lensa Penyimpanan, selama dasbor berada di Akun AWS dan Wilayah asal yang sama. Anda dapat membuat hingga 50 grup Lensa Penyimpanan per Wilayah asal di masing-masing Akun AWS.

Anda dapat membuat dan mengelola grup Lensa Penyimpanan S3 dengan menggunakan konsol Amazon S3 AWS Command Line Interface, (AWS CLI), SDK AWS, atau dengan API REST Amazon S3.

Topik

- Melihat metrik grup Lensa Penyimpanan yang digabungkan
- Izin grup Lensa Penyimpanan
- Konfigurasi grup Lensa Penyimpanan
- tag sumber daya AWS
- Ekspor metrik grup Lensa Penyimpanan

Melihat metrik grup Lensa Penyimpanan yang digabungkan

Anda dapat melihat metrik yang digabungkan untuk grup Lensa Penyimpanan dengan melampirkan grup ke dasbor. Grup Lensa Penyimpanan yang ingin dilampirkan harus berada di Wilayah asal yang ditunjuk di akun dasbor.

Untuk melampirkan grup Lensa Penyimpanan ke dasbor, Anda harus menentukan grup di bagian Agregasi grup Lensa Penyimpanan pada konfigurasi dasbor. Jika Anda memiliki beberapa grup Lensa Penyimpanan, Anda dapat memfilter hasil Agregasi grup Lensa Penyimpanan untuk menyertakan atau mengecualikan hanya grup yang diinginkan saja. Untuk informasi selengkapnya tentang melampirkan grup ke dasbor, lihat the section called “Pasang atau lepaskan grup Lensa Penyimpanan”.

Setelah melampirkan grup, Anda akan melihat data agregasi grup Lensa Penyimpanan tambahan di dasbor dalam waktu 48 jam.

> **Note**
>
> Untuk melihat metrik yang digabungkan untuk grup Lensa Penyimpanan, Anda harus melampirkan grup ke dasbor Lensa Penyimpanan S3.

Izin grup Lensa Penyimpanan

Grup Lensa Penyimpanan memerlukan izin tertentu di (IAM) AWS Identity and Access Management untuk mengotorisasi akses ke tindakan grup Lensa Penyimpanan S3. Untuk memberikan izin tersebut, Anda dapat menggunakan kebijakan IAM berbasis identitas. Anda dapat melampirkan kebijakan ini ke pengguna, grup, atau peran IAM untuk diberikan izin. Izin tersebut dapat mencakup kemampuan untuk membuat atau menghapus grup Lensa Penyimpanan, melihat konfigurasinya, atau mengelola tag.

Pengguna atau peran AM yang diberikan izin harus dimiliki oleh akun yang membuat atau memiliki grup Lensa Penyimpanan.

Untuk dapat menggunakan grup Lensa Penyimpanan dan untuk melihat metrik grup Lensa Penyimpanan, Anda harus terlebih dahulu memiliki izin yang sesuai untuk menggunakan Lensa Penyimpanan S3. Untuk informasi selengkapnya, lihat the section called “Izin Lensa Penyimpanan S3”.

Untuk membuat dan mengelola grup Lensa Penyimpanan S3, Anda harus memiliki izin IAM berikut, tergantung pada tindakan yang ingin dilakukan:

Tindakan	Izin IAM
Buat grup Lensa Penyimpanan baru	`s3:CreateStorageLensGroup`

Tindakan	Izin IAM
Buat grup Lensa Penyimpanan baru dengan tag	`s3:CreateStorageLensGroup` , `s3:TagResource`
Perbarui grup Lensa Penyimpanan yang sudah ada	`s3:UpdateStorageLensGroup`
Tampilkan detail konfigurasi grup Lensa Penyimpanan	`s3:GetStorageLensGroup`
Cantumkan semua grup Lensa Penyimpanan di Wilayah asal	`s3:ListStorageLensGroups`
Hapus grup Lensa Penyimpanan	`s3:DeleteStorageLensGroup`
Cantumkan daftar tag yang ditambahkan ke grup Lensa Penyimpanan	`s3:ListTagsForResource`
Tambahkan atau perbarui tag grup Lensa Penyimpanan untuk grup Lensa Penyimpanan yang sudah ada	`s3:TagResource`
Hapus tag dari grup Lensa Penyimpanan	`s3:UntagResource`

Berikut adalah contoh cara mengonfigurasi kebijakan IAM di akun yang membuat grup Lensa Penyimpanan. Untuk menggunakan kebijakan ini, ganti *us-east-1* dengan Wilayah asal tempat grup Lensa Penyimpanan Anda berada. Ganti *111122223333* dengan ID Akun AWS Anda, dan ganti *example-storage-lens-group* dengan nama grup Lensa Penyimpanan Anda. Untuk menerapkan izin ini ke semua grup Lensa Penyimpanan, ganti *example-storage-lens-group* dengan *.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EXAMPLE-Statement-ID",
            "Effect": "Allow",
            "Action": [
```

```
                "s3:CreateStorageLensGroup",
                "s3:UpdateStorageLensGroup",
                "s3:GetStorageLensGroup",
                "s3:ListStorageLensGroups",
                "s3:DeleteStorageLensGroup,
                "s3:TagResource",
                "s3:UntagResource",
                "s3:ListTagsForResource"
                ],
            "Resource": "arn:aws:s3:us-east-1:111122223333:storage-lens-group/example-
storage-lens-group"
        }
    ]
}
```

Untuk informasi selengkapnya tentang izin Lensa Penyimpanan S3, lihat Izin Lensa Penyimpanan Amazon S3. Untuk informasi selengkapnya tentang bahasa kebijakan IAM, lihat Kebijakan dan izin di Amazon S3.

Konfigurasi grup Lensa Penyimpanan

Nama grup Lensa Penyimpanan S3

Sebaiknya berikan nama grup Lensa Penyimpanan sesuai dengan tujuannya sehingga Anda dapat dengan mudah menentukan grup mana yang ingin dilampirkan ke dasbor. Untuk dapat melampirkan grup Lensa Penyimpanan ke dasbor, Anda harus menentukan grup di bagian agregasi grup Lensa Penyimpanan pada konfigurasi dasbor.

Nama grup Lensa Penyimpanan harus unik di dalam akun tersebut. Nama tidak boleh melebihi 64 karakter, dan dapat berisi huruf (a-z, A-Z), angka (0-9), tanda hubung (-), dan garis bawah (_).

Wilayah Asal

Wilayah asal adalah Wilayah AWS tempat grup Lensa Penyimpanan Anda dibuat dan dipelihara. Grup Lensa Penyimpanan Anda dibuat di Wilayah asal yang sama dengan dasbor Lensa Penyimpanan Amazon S3 Anda. Konfigurasi dan metrik grup Lensa Penyimpanan juga disimpan di Wilayah ini. Anda dapat membuat hingga 50 grup Lensa Penyimpanan di Wilayah asal.

Setelah Anda membuat grup Lensa Penyimpanan, Anda tidak dapat mengedit Wilayah asal.

Cakupan

Untuk menyertakan objek di dalam grup Lensa Penyimpanan, objek tersebut harus berada di cakupan dasbor Lensa Penyimpanan Amazon S3. Cakupan dasbor Lensa Penyimpanan ditentukan oleh bucket yang disertakan di dalam Cakupan dasbor untuk konfigurasi dasbor Lensa Penyimpanan S3 Anda.

Anda dapat menggunakan filter yang berbeda untuk objek guna menentukan cakupan grup Lensa Penyimpanan. Untuk melihat metrik grup Lensa Penyimpanan ini di dasbor Lensa Penyimpanan S3, objek harus cocok dengan filter yang disertakan di dalam grup Lensa Penyimpanan. Misalnya, grup Lensa Penyimpanan Anda menyertakan objek dengan awalan `marketing` dan akhiran `.png`, tetapi tidak ada objek yang cocok dengan kriteria tersebut. Dalam hal ini, metrik untuk grup Lensa Penyimpanan tersebut tidak akan dihasilkan di dalam ekspor metrik harian Anda, dan tidak ada metrik untuk grup ini yang akan terlihat di dasbor.

Filter

Anda dapat menggunakan filter berikut ke dalam grup Lensa Penyimpanan S3:

- Awalan – Menentukan awalan objek yang disertakan, yang merupakan string karakter di awal nama kunci objek tersebut. Misalnya, nilai `images` untuk filter Awalan akan mencakup objek dengan salah satu awalan berikut: `images/`, `images-marketing`, dan `images/production`. Panjang maksimum awalan adalah 1.024 byte.
- Akhiran - Menentukan akhiran objek yang disertakan (misalnya, `.png`, `.jpeg`, atau `.csv`). Panjang maksimum akhiran adalah 1.024 byte.
- Tag objek – Menentukan daftar tag objek yang ingin difilter. Kunci tag tidak boleh melebihi 128 karakter Unicode, dan nilai tag tidak boleh melebihi 256 karakter Unicode. Perhatikan bahwa jika bidang nilai tag objek dibiarkan kosong, grup Lensa Penyimpanan S3 hanya mencocokkan objek dengan objek lain yang juga memiliki nilai tag kosong.
- Usia - Menentukan rentang usia objek dari objek yang disertakan dalam hitungan hari. Hanya bilangan bulat yang didukung.
- Ukuran – Menentukan rentang ukuran objek dari objek disertakan dalam hitungan byte. Hanya bilangan bulat yang didukung. Nilai maksimum yang diperbolehkan adalah 5 TB.

Tag objek grup Lensa Penyimpanan

Anda dapat membuat grup Lensa Penyimpanan yang menyertakan hingga 10 filter tag objek. Contoh berikut mencakup dua pasangan nilai-kunci tag objek sebagai filter untuk grup Lensa Penyimpanan

yang dinamai *Marketing-Department*. Untuk menggunakan contoh ini, ganti *Marketing-Department* dengan nama grup Anda, dan ganti *object-tag-key-1*, *object-tag-value-1*, dan seterusnya dengan pasangan nilai kunci tag objek yang ingin difilter.

```
{
    "Name": "Marketing-Department",
    "Filter": {
     "MatchAnyTag":[
                {
                    "Key": "object-tag-key-1",
                    "Value": "object-tag-value-1"
                },
                {
                    "Key": "object-tag-key-2",
                    "Value": "object-tag-value-2"
                }
            ]
    }
}
```

Operator logis (**`And`** atau **`Or`**)

Untuk menyertakan beberapa kondisi filter di grup Lensa Penyimpanan, Anda dapat menggunakan operator logis (`And` atau `Or`). Di contoh berikut, grup Lensa Penyimpanan yang dinamai *Marketing-Department* memiliki operator `And` yang berisi filter `Prefix`, `ObjectAge`, dan `ObjectSize`. Karena operator `And` digunakan, hanya objek yang cocok dengan semua syarat filter ini yang akan disertakan di dalam cakupan grup Lensa Penyimpanan.

Untuk menggunakan contoh ini, ganti *user input placeholders* dengan nilai yang ingin difilter.

```
{
    "Name": "Marketing-Department",
    "Filter": {
        "And": {
            "MatchAnyPrefix": [
                "prefix-1",
                "prefix-2",
                "prefix-3/sub-prefix-1"
            ],
             "MatchObjectAge": {
                "DaysGreaterThan": 10,
                "DaysLessThan": 60
```

```
            },
            "MatchObjectSize": {
                "BytesGreaterThan": 10,
                "BytesLessThan": 60
            }
        }
    }
}
```

> **Note**
>
> Jika Anda ingin menyertakan objek yang cocok dengan salah satu syarat filter, ganti operator logis `And` dengan operator logis `Or` dalam contoh ini.

tag sumber daya AWS

Setiap grup Lensa Penyimpanan S3 dihitung sebagai sumber daya AWS dengan Amazon Resource Name (ARN) sendiri. Oleh karena itu, Anda mengonfigurasi grup Lensa Penyimpanan, Anda dapat menambahkan tag sumber daya AWS ke grup secara opsional. Anda dapat menambahkan hingga 50 tag untuk setiap grup Lensa Penyimpanan. Untuk membuat grup Lensa Penyimpanan dengan tag, Anda harus memiliki izin `s3:CreateStorageLensGroup` dan `s3:TagResource`.

Anda dapat menggunakan tag sumber daya AWS untuk mengkategorikan sumber daya berdasarkan departemen, lini bisnis, atau proyek. Langkah ini berguna ketika Anda memiliki banyak sumber daya dengan jenis yang sama. Dengan menerapkan tag, Anda dapat dengan cepat mengidentifikasi grup Lensa Penyimpanan tertentu berdasarkan tag yang telah ditetapkan. Anda juga dapat menggunakan tag untuk melacak dan mengalokasikan biaya.

Selain itu, saat menambahkan tag sumber daya AWS ke grup Lensa Penyimpanan, Anda dapat mengaktifkan kontrol akses berbasis atribut (ABAC). ABAC adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut, dalam hal ini tag. Anda juga dapat menggunakan kondisi yang menentukan tag sumber daya ke dalam kebijakan IAM untuk mengontrol akses ke sumber daya AWS.

Anda dapat mengedit kunci dan nilai tanda, dan Anda dapat menghapus tanda dari sumber daya kapan pun. Perhatikan juga batasan berikut ini:

- Kunci tag dan nilai tag peka huruf besar dan kecil.

- Jika Anda menambahkan tanda yang memiliki kunci yang sama dengan tanda yang sudah ada di sumber daya tersebut, nilai baru akan menimpa nilai lama.
- Jika sumber daya dihapus, semua tanda untuk sumber daya tersebut juga akan dihapus.
- Jangan sertakan data pribadi atau sensitif di dalam tag sumber daya AWS.
- Tag sistem (atau tag dengan kunci tag yang dimulai dengan `aws:`) tidak didukung.
- Panjang setiap kunci tag tidak boleh melebihi 128 karakter. Panjang setiap nilai tag tidak boleh melebihi 256 karakter.

Ekspor metrik grup Lensa Penyimpanan

Metrik grup Lensa Penyimpanan S3 disertakan di dalam ekspor metrik Lensa Penyimpanan Amazon S3 untuk dasbor tempat grup Lensa Penyimpanan dilampirkan. Untuk informasi umum tentang fitur ekspor metrik Lensa Penyimpanan, lihat Melihat metrik Lensa Penyimpanan Amazon S3 menggunakan ekspor data.

Ekspor metrik untuk grup Lensa Penyimpanan mencakup metrik Lensa Penyimpanan S3 yang berada di dalam cakupan dasbor tempat grup Lensa Penyimpanan dilampirkan. Ekspor ini juga mencakup data metrik tambahan untuk grup Lensa Penyimpanan.

Setelah membuat grup Lensa Penyimpanan, ekspor metrik akan dikirim setiap hari ke bucket yang dipilih saat Anda mengonfigurasi ekspor metrik untuk dasbor tempat grup dilampirkan. Anda memerlukan waktu hingga 48 jam untuk dapat menerima ekspor metrik pertama.

Untuk menghasilkan metrik di ekspor harian, objek harus cocok dengan filter yang disertakan di dalam grup Lensa Penyimpanan. Jika tidak ada objek yang cocok dengan filter yang disertakan di dalam grup Lensa Penyimpanan, tidak ada metrik yang akan dihasilkan. Namun, jika suatu objek cocok dengan dua atau lebih grup Lensa Penyimpanan, objek tersebut akan dicantumkan secara terpisah untuk setiap grup saat muncul di ekspor metrik.

Anda dapat mengidentifikasi metrik untuk grup Lensa Penyimpanan dengan mencari salah satu nilai berikut di kolom `record_type` ekspor metrik untuk dasbor:

- `STORAGE_LENS_GROUP_BUCKET`
- `STORAGE_LENS_GROUP_ACCOUNT`

Kolom `record_value` menampilkan ARN sumber daya untuk grup Lensa Penyimpanan (misalnya, `arn:aws:s3:`*`us-east-1`*`:`*`111122223333`*`:storage-lens-group/`*`Marketing-Department`*).

Menggunakan grup Lensa Penyimpanan

Grup Amazon S3 Storage Lens menggabungkan metrik menggunakan filter khusus berdasarkan metadata objek. Anda dapat menganalisis dan memfilter metrik Lensa Penyimpanan S3 menggunakan awalan, sufiks, tag objek, ukuran objek, atau usia objek. Dengan grup Lensa Penyimpanan Amazon S3, Anda juga dapat mengkategorikan penggunaan di dalam dan di seluruh bucket Amazon S3. Hasilnya, Anda akan dapat lebih memahami dan mengoptimalkan penyimpanan S3 Anda.

Untuk mulai memvisualisasikan data untuk grup Lensa Penyimpanan, Anda harus terlebih dahulu melampirkan grup Lensa Penyimpanan Anda ke dasbor Lensa Penyimpanan S3. Jika Anda perlu mengelola grup Lensa Penyimpanan di dasbor, Anda dapat mengedit konfigurasi dasbor. Untuk memeriksa grup Lensa Penyimpanan mana yang berada di bawah akun Anda, Anda dapat mencantumkannya. Untuk memeriksa grup Lensa Penyimpanan mana yang dilampirkan ke dasbor Anda, Anda selalu dapat memeriksa tab grup Lensa Penyimpanan di dasbor. Untuk meninjau atau memperbarui cakupan grup Lensa Penyimpanan yang ada, Anda dapat melihat detailnya. Anda juga dapat menghapus grup Lensa Penyimpanan secara permanen.

Untuk mengelola izin, Anda dapat membuat dan menambahkan tag AWS sumber daya yang ditentukan pengguna ke grup Lensa Penyimpanan Anda. Anda dapat menggunakan tag AWS sumber daya untuk mengkategorikan sumber daya menurut departemen, lini bisnis, atau proyek. Melakukannya berguna ketika Anda memiliki banyak sumber daya dari jenis yang sama. Dengan menerapkan tag, Anda dapat dengan cepat mengidentifikasi grup Lensa Penyimpanan tertentu berdasarkan tag yang telah Anda tetapkan padanya.

Selain itu, saat menambahkan tag AWS sumber daya ke grup Lensa Penyimpanan, Anda mengaktifkan kontrol akses berbasis atribut (ABAC). ABAC adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut, dalam hal ini tag. Anda juga dapat menggunakan kondisi yang menentukan tag sumber daya dalam kebijakan IAM Anda untuk mengontrol akses ke AWS sumber daya.

Topik

- Membuat grup Lensa Penyimpanan
- Memasang atau menghapus grup Lensa Penyimpanan S3 ke atau dari dasbor Anda
- Memvisualisasikan data grup Lensa Penyimpanan Anda
- Memperbarui grup Lensa Penyimpanan
- Mengelola tag AWS sumber daya dengan grup Lensa Penyimpanan

- Daftar semua grup Storage Lens
- Melihat detail grup Lensa Penyimpanan
- Menghapus grup Lensa Penyimpanan

Membuat grup Lensa Penyimpanan

Contoh berikut menunjukkan cara membuat grup Lensa Penyimpanan Amazon S3 dengan menggunakan konsol Amazon S3 AWS Command Line Interface ,AWS CLI(), dan. AWS SDK for Java

Menggunakan konsol S3

Untuk membuat grup Lensa Penyimpanan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di bilah navigasi di bagian atas halaman, pilih nama AWS Wilayah yang saat ini ditampilkan. Selanjutnya, pilih Wilayah yang ingin Anda alihkan.
3. Di panel navigasi bagian kiri, pilih Grup Lensa Penyimpanan.
4. Pilih Buat grup Lensa Penyimpanan.
5. Di bawah Umum, lihat Wilayah Rumah Anda dan masukkan nama grup Lensa Penyimpanan Anda.
6. Di bagian bawah Cakupan, pilih filter yang ingin Anda terapkan ke grup Lensa Penyimpanan. Untuk menerapkan beberapa filter, pilih filter Anda, lalu pilih operator logika AND atau OR.
   - Untuk filter Awalan, pilih Awalan, dan masukkan string awalan. Untuk menambahkan beberapa awalan, pilih Tambah awalan. Untuk menghapus awalan, pilih Hapus di bagian samping awalan yang ingin dihapus.
   - Untuk filter Tag objek, pilih Tag objek, dan masukkan pasangan nilai-kunci untuk objek Anda. Kemudian, pilih Tambah tag. Untuk menghapus tag, pilih Hapus yang ada di bagian samping tag yang ingin dihapus.
   - Untuk filter Akhiran, pilih Akhiran, dan masukkan string akhiran. Untuk menambahkan beberapa akhiran, pilih Tambah akhiran. Untuk menghapus akhiran, pilih Hapus di bagian samping akhiran yang ingin dihapus.

- Untuk filter Usia, tentukan rentang usia objek dalam hitungan hari. Pilih Tentukan usia objek minimum, dan masukkan usia objek minimum. Lalu, pilih Tentukan usia objek maksimum, dan masukkan usia objek maksimum.
- Untuk filter Ukuran, tentukan rentang ukuran objek dan unit pengukurannya. Pilih Tentukan ukuran objek minimum, dan masukkan ukuran objek minimum. Pilih Tentukan ukuran objek maksimum, dan masukkan ukuran objek maksimum.

7. (Opsional) Untuk tag AWS sumber daya, tambahkan pasangan kunci-nilai, lalu pilih Tambah tag.
8. Pilih Buat grup Lensa Penyimpanan.

Menggunakan AWS CLI

Contoh AWS CLI perintah berikut membuat grup Storage Lens. Untuk menggunakan contoh perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control create-storage-lens-group --account-id 111122223333 \
--region us-east-1 --storage-lens-group=file://./marketing-department.json
```

Contoh AWS CLI perintah berikut membuat grup Storage Lens dengan dua tag AWS sumber daya. Untuk menggunakan contoh perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control create-storage-lens-group --account-id 111122223333 \
--region us-east-1 --storage-lens-group=file://./marketing-department.json \
--tags Key=k1,Value=v1 Key=k2,Value=v2
```

Untuk contoh konfigurasi JSON, lihat Konfigurasi grup Lensa Penyimpanan.

Menggunakan AWS SDK for Java

AWS SDK for Java Contoh berikut membuat grup Storage Lens. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

Example — Buat grup Lensa Penyimpanan dengan satu filter

Contoh berikut membuat grup Lensa Penyimpanan dengan nama *Marketing-Department*. Grup ini memiliki filter usia objek yang menentukan rentang usia *30* hingga *90* hari. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
package aws.example.s3control;
```

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.CreateStorageLensGroupRequest;
import software.amazon.awssdk.services.s3control.model.MatchObjectAge;
import software.amazon.awssdk.services.s3control.model.StorageLensGroup;
import software.amazon.awssdk.services.s3control.model.StorageLensGroupFilter;

public class CreateStorageLensGroupWithObjectAge {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            StorageLensGroupFilter objectAgeFilter = StorageLensGroupFilter.builder()
                    .matchObjectAge(MatchObjectAge.builder()
                            .daysGreaterThan(30)
                            .daysLessThan(90)
                            .build())
                    .build();

            StorageLensGroup storageLensGroup = StorageLensGroup.builder()
                    .name(storageLensGroupName)
                    .filter(objectAgeFilter)
                    .build();

            CreateStorageLensGroupRequest createStorageLensGroupRequest =
 CreateStorageLensGroupRequest.builder()
                    .storageLensGroup(storageLensGroup)
                    .accountId(accountId).build();

            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.createStorageLensGroup(createStorageLensGroupRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
```

```
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example — Buat grup Lensa Penyimpanan dengan operator **AND** yang menyertakan beberapa filter

Contoh berikut membuat grup Lensa Penyimpanan dengan nama *Marketing-Department*. Grup ini menggunakan operator AND untuk menunjukkan bahwa objek harus cocok dengan semua kondisi filter. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.CreateStorageLensGroupRequest;
import software.amazon.awssdk.services.s3control.model.MatchObjectAge;
import software.amazon.awssdk.services.s3control.model.MatchObjectSize;
import software.amazon.awssdk.services.s3control.model.S3Tag;
import software.amazon.awssdk.services.s3control.model.StorageLensGroup;
import software.amazon.awssdk.services.s3control.model.StorageLensGroupAndOperator;
import software.amazon.awssdk.services.s3control.model.StorageLensGroupFilter;


public class CreateStorageLensGroupWithAndFilter {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            // Create object tags.
            S3Tag tag1 = S3Tag.builder()
                    .key("object-tag-key-1")
                    .value("object-tag-value-1")
                    .build();
            S3Tag tag2 = S3Tag.builder()
                    .key("object-tag-key-2")
```

```
                    .value("object-tag-value-2")
                    .build();

            StorageLensGroupAndOperator andOperator =
StorageLensGroupAndOperator.builder()
                    .matchAnyPrefix("prefix-1", "prefix-2", "prefix-3/sub-prefix-1")
                    .matchAnySuffix(".png", ".gif", ".jpg")
                    .matchAnyTag(tag1, tag2)
                    .matchObjectAge(MatchObjectAge.builder()
                            .daysGreaterThan(30)
                            .daysLessThan(90).build())
                    .matchObjectSize(MatchObjectSize.builder()
                            .bytesGreaterThan(1000L)
                            .bytesLessThan(6000L).build())
                    .build();

            StorageLensGroupFilter andFilter = StorageLensGroupFilter.builder()
                    .and(andOperator)
                    .build();

            StorageLensGroup storageLensGroup = StorageLensGroup.builder()
                    .name(storageLensGroupName)
                    .filter(andFilter)
                    .build();

            CreateStorageLensGroupRequest createStorageLensGroupRequest =
CreateStorageLensGroupRequest.builder()
                    .storageLensGroup(storageLensGroup)
                    .accountId(accountId).build();

            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.createStorageLensGroup(createStorageLensGroupRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
```

```
    }
}
```

Example — Buat grup Lensa Penyimpanan dengan operator **OR** yang menyertakan beberapa filter

Contoh berikut membuat grup Lensa Penyimpanan dengan nama *Marketing-Department*. Grup ini menggunakan operator OR untuk menerapkan filter awalan (*prefix-1*, *prefix-2*, *prefix3/sub-prefix-1*) atau filter ukuran objek dengan rentang ukuran antara *1000* byte dan *6000* byte. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.CreateStorageLensGroupRequest;
import software.amazon.awssdk.services.s3control.model.MatchObjectSize;
import software.amazon.awssdk.services.s3control.model.StorageLensGroup;
import software.amazon.awssdk.services.s3control.model.StorageLensGroupFilter;
import software.amazon.awssdk.services.s3control.model.StorageLensGroupOrOperator;

public class CreateStorageLensGroupWithOrFilter {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            StorageLensGroupOrOperator orOperator =
 StorageLensGroupOrOperator.builder()
                    .matchAnyPrefix("prefix-1", "prefix-2", "prefix-3/sub-prefix-1")
                    .matchObjectSize(MatchObjectSize.builder()
                            .bytesGreaterThan(1000L)
                            .bytesLessThan(6000L)
                            .build())
                    .build();

            StorageLensGroupFilter orFilter = StorageLensGroupFilter.builder()
                    .or(orOperator)
                    .build();

            StorageLensGroup storageLensGroup = StorageLensGroup.builder()
```

```
                    .name(storageLensGroupName)
                    .filter(orFilter)
                    .build();

            CreateStorageLensGroupRequest createStorageLensGroupRequest =
 CreateStorageLensGroupRequest.builder()
                    .storageLensGroup(storageLensGroup)
                    .accountId(accountId).build();

            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.createStorageLensGroup(createStorageLensGroupRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Example — Buat grup Lensa Penyimpanan dengan satu filter dan dua tag AWS sumber daya

Contoh berikut membuat grup Lensa Penyimpanan dengan nama `Marketing-Department` yang memiliki filter akhiran. Contoh ini juga menambahkan dua tag AWS sumber daya ke grup Storage Lens. Untuk menggunakan contoh ini, ganti `user input placeholders` dengan informasi Anda sendiri.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.CreateStorageLensGroupRequest;
import software.amazon.awssdk.services.s3control.model.StorageLensGroup;
import software.amazon.awssdk.services.s3control.model.StorageLensGroupFilter;
```

```
import software.amazon.awssdk.services.s3control.model.Tag;

public class CreateStorageLensGroupWithResourceTags {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            // Create AWS resource tags.
            Tag resourceTag1 = Tag.builder()
                    .key("resource-tag-key-1")
                    .value("resource-tag-value-1")
                    .build();
            Tag resourceTag2 = Tag.builder()
                    .key("resource-tag-key-2")
                    .value("resource-tag-value-2")
                    .build();

            StorageLensGroupFilter suffixFilter = StorageLensGroupFilter.builder()
                    .matchAnySuffix(".png", ".gif", ".jpg")
                    .build();

            StorageLensGroup storageLensGroup = StorageLensGroup.builder()
                    .name(storageLensGroupName)
                    .filter(suffixFilter)
                    .build();

            CreateStorageLensGroupRequest createStorageLensGroupRequest =
 CreateStorageLensGroupRequest.builder()
                    .storageLensGroup(storageLensGroup)
                    .tags(resourceTag1, resourceTag2)
                    .accountId(accountId).build();

            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.createStorageLensGroup(createStorageLensGroupRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
```

```
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Untuk contoh konfigurasi JSON, lihat Konfigurasi grup Lensa Penyimpanan.

Memasang atau menghapus grup Lensa Penyimpanan S3 ke atau dari dasbor Anda

Setelah memutakhirkan ke tingkat lanjutan di Amazon S3 Storage Lens, Anda dapat melampirkan grup Storage Lens ke dasbor Anda. Jika Anda memiliki beberapa grup Lensa Penyimpanan, Anda dapat menyertakan atau mengecualikan grup yang Anda inginkan.

Grup Lensa Penyimpanan Anda harus berada di dalam Wilayah rumah yang ditunjuk di akun dasbor. Setelah melampirkan grup Lensa Penyimpanan ke dasbor, Anda akan menerima data agregasi grup Lensa Penyimpanan tambahan dalam ekspor metrik Anda dalam waktu 48 jam.

> **Note**
>
> Jika Anda ingin melihat metrik agregat untuk grup Lensa Penyimpanan, Anda harus melampirkannya ke dasbor Lensa Penyimpanan. Untuk contoh file konfigurasi JSON grup Storage Lens, lihatContoh konfigurasi Lensa Penyimpanan S3 dengan grup Lensa Penyimpanan di JSON.

Memasang grup Lensa Penyimpanan ke dasbor Lensa Penyimpanan S3

Untuk melampirkan grup Lensa Penyimpanan ke dasbor Lensa Penyimpanan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, di bawah Lensa Penyimpanan, pilih Dasbor.
3. Pilih tombol opsi untuk dasbor Storage Lens yang ingin Anda lampirkan ke grup Storage Lens.
4. Pilih Edit.
5. Di bawah Pemilihan metrik, pilih Metrik dan rekomendasi lanjutan.
6. Pilih agregasi grup Lensa Penyimpanan.

> **Note**
>
> Secara default, Metrik lanjutan juga dipilih. Namun, Anda juga dapat membatalkan pilihan pengaturan ini karena tidak diperlukan untuk menggabungkan data grup Lensa Penyimpanan.

7. Gulir ke bawah ke agregasi grup Lensa Penyimpanan dan tentukan grup atau grup Lensa Penyimpanan yang ingin Anda sertakan atau kecualikan dalam agregasi data. Anda dapat menggunakan opsi pemfilteran berikut:
   - Jika Anda ingin menyertakan grup Lensa Penyimpanan tertentu, pilih Sertakan grup Lensa Penyimpanan. Di bawah grup Lensa Penyimpanan untuk disertakan, pilih grup Lensa Penyimpanan Anda.
   - Jika Anda ingin menyertakan semua grup Lensa Penyimpanan, pilih Sertakan semua grup Lensa Penyimpanan di wilayah beranda di akun ini.
   - Jika Anda ingin mengecualikan grup Lensa Penyimpanan tertentu, pilih Kecualikan grup Lensa Penyimpanan. Di bawah grup Lensa Penyimpanan untuk dikecualikan, pilih grup Lensa Penyimpanan yang ingin Anda kecualikan.
8. Pilih Save changes (Simpan perubahan). Jika Anda telah mengonfigurasi grup Lensa Penyimpanan dengan benar, Anda akan melihat data agregasi grup Lensa Penyimpanan tambahan di dasbor Anda dalam waktu 48 jam.

Menghapus grup Lensa Penyimpanan dari dasbor Lensa Penyimpanan S3

Untuk menghapus grup Lensa Penyimpanan dari dasbor Lensa Penyimpanan S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, di bawah Lensa Penyimpanan, pilih Dasbor.
3. Pilih tombol opsi untuk dasbor Lensa Penyimpanan tempat Anda ingin menghapus grup Lensa Penyimpanan.
4. Pilih Lihat konfigurasi dasbor.
5. Pilih Edit.
6. Gulir ke bawah ke bagian Pemilihan Metrik.

7. Di bawah agregasi grup Storage Lens, pilih X di sebelah grup Storage Lens yang ingin Anda hapus. Ini menghapus grup Lensa Penyimpanan Anda.

   Jika Anda menyertakan semua grup Lensa Penyimpanan di dasbor, kosongkan kotak centang di samping Sertakan semua grup Lensa Penyimpanan di wilayah beranda di akun ini.

8. Pilih Save changes (Simpan perubahan).

> **Note**
>
> Ini akan memakan waktu hingga 48 jam untuk dasbor Anda untuk mencerminkan pembaruan konfigurasi.

Memvisualisasikan data grup Lensa Penyimpanan Anda

Anda dapat memvisualisasikan data grup Lensa Penyimpanan dengan melampirkan grup ke dasbor Amazon S3 Storage Lens Anda. Setelah Anda menyertakan grup Storage Lens dalam agregasi grup Storage Lens dalam konfigurasi dasbor Anda, data grup Storage Lens dapat memakan waktu hingga 48 jam untuk ditampilkan di dasbor Anda.

Setelah konfigurasi dasbor diperbarui, grup Lensa Penyimpanan yang baru dilampirkan akan muncul di daftar sumber daya yang tersedia di bawah tab grup Lensa Penyimpanan. Anda juga dapat menganalisis lebih lanjut penggunaan penyimpanan di tab Ikhtisar Anda dengan memotong data dengan dimensi lain. Misalnya, Anda dapat memilih salah satu item yang tercantum di bawah 3 kategori Teratas dan memilih Menganalisis dengan untuk mengiris data dengan dimensi lain. Anda tidak dapat menerapkan dimensi yang sama dengan filter itu sendiri.

> **Note**
>
> Anda tidak dapat menerapkan filter grup Lensa Penyimpanan bersama dengan filter awalan, atau sebaliknya. Anda juga tidak dapat menganalisis grup Lensa Penyimpanan lebih lanjut dengan menggunakan filter awalan.

Anda dapat menggunakan tab grup Lensa Penyimpanan di dasbor Amazon S3 Storage Lens untuk menyesuaikan visualisasi data untuk grup Lensa Penyimpanan yang dilampirkan ke dasbor Anda. Anda dapat memvisualisasikan data untuk beberapa grup Lensa Penyimpanan yang dilampirkan ke dasbor Anda, atau semuanya.

Saat memvisualisasikan data grup Lensa Penyimpanan di dasbor Lensa Penyimpanan S3 Anda, perhatikan hal-hal berikut:

- S3 Storage Lens menggabungkan metrik penggunaan untuk objek di bawah semua grup Lensa Penyimpanan yang cocok. Oleh karena itu, jika suatu objek cocok dengan kondisi filter untuk dua atau lebih grup Lensa Penyimpanan, Anda akan melihat penghitungan berulang untuk objek yang sama di seluruh penggunaan penyimpanan Anda.
- Objek harus cocok dengan filter yang Anda sertakan dalam grup Lensa Penyimpanan Anda. Jika tidak ada objek yang cocok dengan filter yang Anda sertakan dalam grup Lensa Penyimpanan, maka tidak ada metrik yang dihasilkan. Untuk menentukan apakah ada objek yang belum ditetapkan, periksa jumlah total objek Anda di dasbor di tingkat akun dan tingkat bucket.

Memperbarui grup Lensa Penyimpanan

Contoh berikut menunjukkan cara memperbarui grup Lensa Penyimpanan Amazon S3. Anda dapat memperbarui grup Lensa Penyimpanan menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), dan. AWS SDK for Java

Menggunakan konsol S3

Untuk memperbarui grup Lensa Penyimpanan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih grup Lensa Penyimpanan.
3. Di bawah grup Lensa Penyimpanan, pilih grup Lensa Penyimpanan yang ingin Anda perbarui.
4. Di bawah Lingkup, pilih Edit.
5. Pada halaman Lingkup, pilih filter yang ingin Anda terapkan ke grup Lensa Penyimpanan Anda. Untuk menerapkan beberapa filter, pilih filter Anda, dan pilih operator logis AND atau OR.
    - Untuk filter Awalan, pilih Awalan, dan masukkan string awalan. Untuk menambahkan beberapa awalan, pilih Tambah awalan. Untuk menghapus awalan, pilih Hapus di samping awalan yang ingin Anda hapus.
    - Untuk filter tag Object, masukkan pasangan kunci-nilai untuk objek Anda. Kemudian, pilih Tambah tag. Untuk menghapus tag, pilih Hapus di samping tag yang ingin Anda hapus.

- Untuk filter Sufiks, pilih Sufiks, dan masukkan string akhiran. Untuk menambahkan beberapa sufiks, pilih Tambahkan akhiran. Untuk menghapus akhiran, pilih Hapus di sebelah akhiran yang ingin Anda hapus.
- Untuk filter Usia, tentukan rentang usia objek dalam hari. Pilih Tentukan usia objek minimum, dan masukkan usia objek minimum. Untuk Tentukan usia objek maksimum, masukkan usia objek maksimum.
- Untuk filter Ukuran, tentukan rentang ukuran objek dan unit pengukuran. Pilih Tentukan ukuran objek minimum, dan masukkan ukuran objek minimum. Untuk Tentukan ukuran objek maksimum, masukkan ukuran objek maksimum.

6. Pilih Save changes (Simpan perubahan). Halaman detail untuk grup Storage Lens muncul.
7. (Opsional) Jika Anda ingin menambahkan tag AWS sumber daya baru, gulir ke bagian tag AWS sumber daya, lalu pilih Tambahkan tag. Halaman Tambahkan penandaan akan muncul.

   Tambahkan pasangan kunci-nilai baru, lalu pilih Simpan perubahan. Halaman detail untuk grup Storage Lens muncul.
8. (Opsional) Jika Anda ingin menghapus tag AWS sumber daya yang ada, gulir ke bagian tag AWS sumber daya, dan pilih tag sumber daya. Lalu, pilih Hapus. Kotak dialog Hapus AWS tag muncul.

   Pilih Hapus lagi untuk menghapus tag AWS sumber daya secara permanen.

   > **Note**
   >
   > Setelah Anda menghapus tag AWS sumber daya secara permanen, tag tersebut tidak dapat dipulihkan.

Menggunakan AWS CLI

AWS CLIContoh perintah berikut mengembalikan rincian konfigurasi untuk grup Storage Lens bernama*marketing-department*. Untuk menggunakan perintah contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control get-storage-lens-group --account-id 111122223333 \
--region us-east-1 --name marketing-department
```

AWS CLIContoh berikut memperbarui grup Storage Lens. Untuk menggunakan perintah contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control update-storage-lens-group --account-id 111122223333 \
--region us-east-1 --storage-lens-group=file://./marketing-department.json
```

Misalnya konfigurasi JSON, lihat. Konfigurasi grup Lensa Penyimpanan

Menggunakan AWS SDK for Java

AWS SDK for JavaContoh berikut mengembalikan detail konfigurasi untuk grup *Marketing-Department* Storage Lens di akun*111122223333*. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.GetStorageLensGroupRequest;
import software.amazon.awssdk.services.s3control.model.GetStorageLensGroupResponse;

public class GetStorageLensGroup {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            GetStorageLensGroupRequest getRequest =
 GetStorageLensGroupRequest.builder()
                    .name(storageLensGroupName)
                    .accountId(accountId).build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            GetStorageLensGroupResponse response =
 s3ControlClient.getStorageLensGroup(getRequest);
            System.out.println(response);
        } catch (AmazonServiceException e) {
```

```
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Contoh berikut memperbarui grup Storage Lens bernama *Marketing-Department* dalam akun*111122223333*. Contoh ini memperbarui cakupan dasbor untuk menyertakan objek yang cocok dengan salah satu sufiks berikut:*.png*,, *.gif.jpg*, atau. *.jpeg* Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.StorageLensGroup;
import software.amazon.awssdk.services.s3control.model.StorageLensGroupFilter;
import software.amazon.awssdk.services.s3control.model.UpdateStorageLensGroupRequest;

public class UpdateStorageLensGroup {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            // Create updated filter.
            StorageLensGroupFilter suffixFilter = StorageLensGroupFilter.builder()
                    .matchAnySuffix(".png", ".gif", ".jpg", ".jpeg")
                    .build();

            StorageLensGroup storageLensGroup = StorageLensGroup.builder()
                    .name(storageLensGroupName)
                    .filter(suffixFilter)
                    .build();
```

```
            UpdateStorageLensGroupRequest updateStorageLensGroupRequest =
 UpdateStorageLensGroupRequest.builder()
                    .name(storageLensGroupName)
                    .storageLensGroup(storageLensGroup)
                    .accountId(accountId)
                    .build();

            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.updateStorageLensGroup(updateStorageLensGroupRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Misalnya konfigurasi JSON, lihat. Konfigurasi grup Lensa Penyimpanan

Mengelola tag AWS sumber daya dengan grup Lensa Penyimpanan

Setiap grup Lensa Penyimpanan Amazon S3 dihitung sebagai AWS sumber daya dengan Nama Sumber Daya Amazon (ARN) miliknya sendiri. Oleh karena itu, saat Anda mengonfigurasi grup Lensa Penyimpanan, Anda dapat menambahkan tag AWS sumber daya ke grup secara opsional. Anda dapat menambahkan hingga 50 tag untuk setiap grup Lensa Penyimpanan. Untuk membuat grup Lensa Penyimpanan dengan tag, Anda harus memiliki `s3:CreateStorageLensGroup` dan `s3:TagResource` izin.

Anda dapat menggunakan tag AWS sumber daya untuk mengkategorikan sumber daya menurut departemen, lini bisnis, atau proyek. Melakukannya berguna ketika Anda memiliki banyak sumber daya dari jenis yang sama. Dengan menerapkan tag, Anda dapat dengan cepat mengidentifikasi grup Lensa Penyimpanan tertentu berdasarkan tag yang telah Anda tetapkan padanya. Anda juga dapat menggunakan tag untuk melacak dan mengalokasikan biaya.

Selain itu, saat menambahkan tag AWS sumber daya ke grup Lensa Penyimpanan, Anda mengaktifkan kontrol akses berbasis atribut (ABAC). ABAC adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut, dalam hal ini tag. Anda juga dapat menggunakan kondisi yang menentukan tag sumber daya dalam kebijakan IAM Anda untuk mengontrol akses ke AWS sumber daya.

Anda dapat mengedit kunci dan nilai tanda, dan Anda dapat menghapus tanda dari sumber daya pada saat kapan pun. Juga, waspadai batasan berikut:

- Kunci tag dan nilai tag peka huruf besar/kecil.
- Jika Anda menambahkan tanda yang memiliki kunci yang sama dengan tanda yang telah ada pada sumber daya tersebut, nilai yang baru akan menimpa nilai yang lama.
- Jika Anda menghapus sumber daya, semua tanda untuk sumber daya tersebut juga dihapus.
- Jangan sertakan data pribadi atau sensitif dalam tag AWS sumber daya Anda.
- Tag sistem (dengan kunci tag yang dimulai dengan`aws:`) tidak didukung.
- Panjang setiap tombol tag tidak boleh melebihi 128 karakter. Panjang setiap nilai tag tidak boleh melebihi 256 karakter.

Contoh berikut menunjukkan cara menggunakan tag AWS sumber daya dengan grup Storage Lens.

Topik

- Menambahkan tag AWS sumber daya ke grup Lensa Penyimpanan
- Memperbarui nilai tag grup Lensa Penyimpanan
- Menghapus tag AWS sumber daya dari grup Lensa Penyimpanan
- Daftar tag grup Lensa Penyimpanan

Menambahkan tag AWS sumber daya ke grup Lensa Penyimpanan

Contoh berikut menunjukkan cara menambahkan tag AWS sumber daya ke grup Lensa Penyimpanan Amazon S3. Anda dapat menambahkan tag sumber daya dengan menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), dan. AWS SDK for Java

Menggunakan konsol S3

Untuk menambahkan tag AWS sumber daya ke grup Lensa Penyimpanan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih grup Lensa Penyimpanan.
3. Di bawah grup Lensa Penyimpanan, pilih grup Lensa Penyimpanan yang ingin Anda perbarui.
4. Di bawah tag AWS sumber daya, pilih Tambahkan tag.
5. Pada halaman Tambahkan tag, tambahkan pasangan kunci-nilai baru.

> **Note**
>
> Menambahkan tag baru dengan kunci yang sama dengan tag yang ada menimpa nilai tag sebelumnya.

6. (Opsional) Untuk menambahkan lebih dari satu tag baru, pilih Tambah tag lagi untuk melanjutkan menambahkan entri baru. Anda dapat menambahkan hingga 50 tag AWS sumber daya ke grup Lensa Penyimpanan Anda.
7. (Opsional) Jika Anda ingin menghapus entri yang baru ditambahkan, pilih Hapus di samping tag yang ingin Anda hapus.
8. Pilih Save changes (Simpan perubahan).

Menggunakan perintah AWS CLI

AWS CLIPerintah contoh berikut menambahkan dua tag sumber daya ke grup Lensa Penyimpanan yang ada bernama*marketing-department*. Untuk menggunakan perintah contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control tag-resource --account-id 111122223333 \
--resource-arn arn:aws:s3:us-east-1:111122223333:storage-lens-group/marketing-
department \
--region us-east-1 --tags Key=k1,Value=v1 Key=k2,Value=v2
```

Menggunakan AWS SDK for Java

AWS SDK for JavaContoh berikut menambahkan dua tag AWS sumber daya ke grup Lensa Penyimpanan yang ada. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.Tag;
import software.amazon.awssdk.services.s3control.model.TagResourceRequest;

public class TagResource {
    public static void main(String[] args) {
        String resourceARN = "Resource_ARN";
        String accountId = "111122223333";

        try {
            Tag resourceTag1 = Tag.builder()
                .key("resource-tag-key-1")
                .value("resource-tag-value-1")
                .build();
            Tag resourceTag2 = Tag.builder()
                    .key("resource-tag-key-2")
                    .value("resource-tag-value-2")
                    .build();
            TagResourceRequest tagResourceRequest = TagResourceRequest.builder()
                    .resourceArn(resourceARN)
                    .tags(resourceTag1, resourceTag2)
                    .accountId(accountId)
                    .build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.tagResource(tagResourceRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
```

```
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Memperbarui nilai tag grup Lensa Penyimpanan

Contoh berikut menunjukkan cara memperbarui nilai tag grup Storage Lens dengan menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), dan. AWS SDK for Java

Menggunakan konsol S3

Untuk memperbarui tag AWS sumber daya untuk grup Lensa Penyimpanan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih grup Lensa Penyimpanan.
3. Di bawah grup Lensa Penyimpanan, pilih grup Lensa Penyimpanan yang ingin Anda perbarui.
4. Di bawah tag AWS sumber daya, pilih tag yang ingin Anda perbarui.
5. Tambahkan nilai tag baru, menggunakan kunci yang sama dari pasangan kunci-nilai yang ingin Anda perbarui. Pilih ikon tanda centang untuk memperbarui nilai tag.

   > **Note**
   >
   > Menambahkan tag baru dengan kunci yang sama dengan tag yang ada menimpa nilai tag sebelumnya.

6. (Opsional) Jika Anda ingin menambahkan tag baru, pilih Tambahkan tag untuk menambahkan entri baru. Halaman Tambahkan penandaan akan muncul.

   Anda dapat menambahkan hingga 50 tag AWS sumber daya untuk grup Lensa Penyimpanan Anda. Setelah selesai menambahkan tag baru, pilih Simpan perubahan.
7. (Opsional) Jika Anda ingin menghapus entri yang baru ditambahkan, pilih Hapus di samping tag yang ingin Anda hapus. Setelah selesai menghapus tag, pilih Simpan perubahan.

Menggunakan AWS CLI

Contoh AWS CLI perintah berikut memperbarui dua nilai tag untuk grup Storage Lens bernama*marketing-department*. Untuk menggunakan perintah contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control tag-resource --account-id 111122223333 \
--resource-arn arn:aws:s3:us-east-1:111122223333:storage-lens-group/marketing-
department \
--region us-east-1 --tags Key=k1,Value=v3 Key=k2,Value=v4
```

Menggunakan AWS SDK for Java

AWS SDK for JavaContoh berikut memperbarui dua nilai tag grup Storage Lens. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.Tag;
import software.amazon.awssdk.services.s3control.model.TagResourceRequest;

public class UpdateTagsForResource {
    public static void main(String[] args) {
        String resourceARN = "Resource_ARN";
        String accountId = "111122223333";

        try {
            Tag updatedResourceTag1 = Tag.builder()
                .key("resource-tag-key-1")
                .value("resource-tag-updated-value-1")
                .build();
            Tag updatedResourceTag2 = Tag.builder()
                    .key("resource-tag-key-2")
                    .value("resource-tag-updated-value-2")
                    .build();
            TagResourceRequest tagResourceRequest = TagResourceRequest.builder()
                    .resourceArn(resourceARN)
```

```
                    .tags(updatedResourceTag1, updatedResourceTag2)
                    .accountId(accountId)
                    .build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.tagResource(tagResourceRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Menghapus tag AWS sumber daya dari grup Lensa Penyimpanan

Contoh berikut menunjukkan cara menghapus tag AWS sumber daya dari grup Lensa Penyimpanan. Anda dapat menghapus tag dengan menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), dan. AWS SDK for Java

Menggunakan konsol S3

Untuk menghapus tag AWS sumber daya dari grup Lensa Penyimpanan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih grup Lensa Penyimpanan.
3. Di bawah grup Lensa Penyimpanan, pilih grup Lensa Penyimpanan yang ingin Anda perbarui.
4. Di bawah tag AWS sumber daya, pilih pasangan kunci-nilai yang ingin Anda hapus.
5. Pilih Delete (Hapus). Kotak dialog Hapus tag AWS sumber daya muncul.

> **Note**
>
> Jika tag digunakan untuk mengontrol akses, melanjutkan tindakan ini dapat memengaruhi sumber daya terkait. Setelah Anda menghapus tag secara permanen, tag tidak dapat dipulihkan.

6. Pilih Hapus untuk menghapus pasangan kunci-nilai secara permanen.

Menggunakan AWS CLI

AWS CLIPerintah berikut menghapus dua tag AWS sumber daya dari grup Lensa Penyimpanan yang ada: Untuk menggunakan perintah contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control untag-resource --account-id 111122223333 \
--resource-arn arn:aws:s3:us-east-1:111122223333:storage-lens-group/Marketing-
Department \
--region us-east-1 --tag-keys k1 k2
```

Menggunakan AWS SDK for Java

AWS SDK for JavaContoh berikut menghapus dua tag AWS sumber daya dari grup Lensa Penyimpanan Amazon Resource Name (ARN) yang Anda tentukan di akun. *111122223333* Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.UntagResourceRequest;

public class UntagResource {
    public static void main(String[] args) {
        String resourceARN = "Resource_ARN";
        String accountId = "111122223333";

        try {
```

```
            String tagKey1 = "resource-tag-key-1";
            String tagKey2 = "resource-tag-key-2";
            UntagResourceRequest untagResourceRequest = UntagResourceRequest.builder()
                    .resourceArn(resourceARN)
                    .tagKeys(tagKey1, tagKey2)
                    .accountId(accountId)
                    .build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.untagResource(untagResourceRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Daftar tag grup Lensa Penyimpanan

Contoh berikut menunjukkan cara membuat daftar tag AWS sumber daya yang terkait dengan grup Lensa Penyimpanan. Anda dapat mencantumkan tag dengan menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), dan. AWS SDK for Java

Menggunakan konsol S3

Untuk meninjau daftar tag dan nilai tag untuk grup Lensa Penyimpanan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih grup Lensa Penyimpanan.
3. Di bawah grup Lensa Penyimpanan, pilih grup Lensa Penyimpanan yang Anda minati.
4. Gulir ke bawah ke bagian tag AWS sumber daya. Semua tag AWS sumber daya yang ditentukan pengguna yang ditambahkan ke grup Lensa Penyimpanan Anda terdaftar bersama dengan nilai tag mereka.

Menggunakan AWS CLI

AWS CLIContoh perintah berikut mencantumkan semua nilai tag grup Storage Lens untuk grup Storage Lens bernama*marketing-department*. Untuk menggunakan perintah contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control list-tags-for-resource --account-id 111122223333 \
--resource-arn arn:aws:s3:us-east-1:111122223333:storage-lens-group/marketing-
department \
--region us-east-1
```

Menggunakan AWS SDK for Java

AWS SDK for JavaContoh berikut mencantumkan nilai tag grup Lensa Penyimpanan untuk grup Lensa Penyimpanan Amazon Resource Name (ARN) yang Anda tentukan. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.ListTagsForResourceRequest;
import software.amazon.awssdk.services.s3control.model.ListTagsForResourceResponse;

public class ListTagsForResource {
    public static void main(String[] args) {
        String resourceARN = "Resource_ARN";
        String accountId = "111122223333";

        try {
            ListTagsForResourceRequest listTagsForResourceRequest =
 ListTagsForResourceRequest.builder()
                    .resourceArn(resourceARN)
                    .accountId(accountId)
                    .build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
```

```
            ListTagsForResourceResponse response =
 s3ControlClient.listTagsForResource(listTagsForResourceRequest);
            System.out.println(response);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Daftar semua grup Storage Lens

Contoh berikut menunjukkan cara membuat daftar semua grup Lensa Penyimpanan Amazon S3 di Wilayah Akun AWS dan rumah. Contoh-contoh ini menunjukkan caranya daftar semua grup Storage Lens dengan menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), dan. AWS SDK for Java

Menggunakan konsol S3

Untuk mencantumkan semua grup Lensa Penyimpanan di akun dan beranda Wilayah

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih grup Lensa Penyimpanan.
3. Di bawah grup Lensa Penyimpanan, daftar grup Lensa Penyimpanan di akun Anda akan ditampilkan.

Menggunakan AWS CLI

AWS CLIContoh berikut mencantumkan semua grup Lensa Penyimpanan untuk akun Anda. Untuk menggunakan perintah contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control list-storage-lens-groups --account-id 111122223333 \
--region us-east-1
```

Menggunakan AWS SDK for Java

AWS SDK for JavaContoh berikut mencantumkan grup Storage Lens untuk akun*111122223333*. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.ListStorageLensGroupsRequest;
import software.amazon.awssdk.services.s3control.model.ListStorageLensGroupsResponse;

public class ListStorageLensGroups {
    public static void main(String[] args) {
        String accountId = "111122223333";

        try {
            ListStorageLensGroupsRequest listStorageLensGroupsRequest =
 ListStorageLensGroupsRequest.builder()
                    .accountId(accountId)
                    .build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            ListStorageLensGroupsResponse response =
 s3ControlClient.listStorageLensGroups(listStorageLensGroupsRequest);
            System.out.println(response);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Melihat detail grup Lensa Penyimpanan

Contoh berikut menunjukkan cara melihat detail konfigurasi grup Lensa Penyimpanan Amazon S3. Anda dapat melihat detail ini dengan menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), dan. AWS SDK for Java

Menggunakan konsol S3

Untuk melihat detail konfigurasi grup Lensa Penyimpanan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih grup Lensa Penyimpanan.
3. Di bawah grup Lensa Penyimpanan, pilih tombol opsi di samping grup Lensa Penyimpanan yang Anda minati.
4. Pilih View details (Lihat detail). Anda sekarang dapat meninjau detail grup Lensa Penyimpanan Anda.

Menggunakan AWS CLI

AWS CLIContoh berikut mengembalikan detail konfigurasi untuk grup Storage Lens. Untuk menggunakan perintah contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control get-storage-lens-group --account-id 111122223333 \
--region us-east-1 --name marketing-department
```

Menggunakan AWS SDK for Java

AWS SDK for JavaContoh berikut mengembalikan detail konfigurasi untuk grup Storage Lens bernama *`Marketing-Department`* dalam akun*`111122223333`*. Untuk menggunakan contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
```

```
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.GetStorageLensGroupRequest;
import software.amazon.awssdk.services.s3control.model.GetStorageLensGroupResponse;

public class GetStorageLensGroup {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            GetStorageLensGroupRequest getRequest =
 GetStorageLensGroupRequest.builder()
                    .name(storageLensGroupName)
                    .accountId(accountId).build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            GetStorageLensGroupResponse response =
 s3ControlClient.getStorageLensGroup(getRequest);
            System.out.println(response);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Menghapus grup Lensa Penyimpanan

Contoh berikut menunjukkan cara menghapus grup Lensa Penyimpanan Amazon S3 dengan menggunakan konsol Amazon S3AWS Command Line Interface, AWS CLI (), dan. AWS SDK for Java

Menggunakan konsol S3

Untuk menghapus grup Lensa Penyimpanan

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih grup Lensa Penyimpanan.
3. Di bawah grup Lensa Penyimpanan, pilih tombol opsi di samping grup Lensa Penyimpanan yang ingin Anda hapus.
4. Pilih Delete (Hapus). Sebuah kotak dialog grup Delete Storage Lens ditampilkan.
5. Pilih Hapus lagi untuk menghapus grup Lensa Penyimpanan Anda secara permanen.

> **Note**
>
> Setelah Anda menghapus grup Lensa Penyimpanan, grup Lensa Penyimpanan tidak dapat dipulihkan.

Menggunakan AWS CLI

AWS CLIContoh berikut menghapus grup Storage Lens bernama*marketing-department*. Untuk menggunakan perintah contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3control delete-storage-lens-group --account-id 111122223333 \
--region us-east-1 --name marketing-department
```

Menggunakan AWS SDK for Java

AWS SDK for JavaContoh berikut menghapus grup Storage Lens bernama *Marketing-Department* dalam akun*111122223333*. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
package aws.example.s3control;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.DeleteStorageLensGroupRequest;

public class DeleteStorageLensGroup {
    public static void main(String[] args) {
        String storageLensGroupName = "Marketing-Department";
        String accountId = "111122223333";

        try {
            DeleteStorageLensGroupRequest deleteStorageLensGroupRequest =
 DeleteStorageLensGroupRequest.builder()
                    .name(storageLensGroupName)
                    .accountId(accountId).build();
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                    .region(Region.US_WEST_2)
                    .credentialsProvider(ProfileCredentialsProvider.create())
                    .build();
            s3ControlClient.deleteStorageLensGroup(deleteStorageLensGroupRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it and returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Melacak permintaan Amazon S3 menggunakan AWS X-Ray

AWS X-Ray mengumpulkan data tentang permintaan yang digunakan aplikasi Anda. Anda kemudian dapat melihat dan mem-filter data untuk mengidentifikasi dan memecahkan masalah performa dan kesalahan dalam aplikasi terdistribusi dan arsitektur layanan mikro. Untuk setiap permintaan yang dilacak ke aplikasi Anda, hal itu menunjukkan kepada Anda informasi yang mendetail tentang permintaan, respons, dan panggilan yang dibuat aplikasi Anda untuk membawa sumber daya AWS, layanan-mikro, basis data dan API web HTTP ke hilir.

Untuk informasi lebih lanjut, lihat Apa yang dimaksud AWS X-Ray? dalam Panduan Developer AWS X-Ray.

Topik

- Cara X-Ray bekerja dengan Amazon S3
- Wilayah yang Tersedia

Cara X-Ray bekerja dengan Amazon S3

AWS X-Ray mendukung perambatan konteks pelacakan untuk Amazon S3, sehingga Anda dapat melihat permintaan end-to-end saat permintaan tersebut melakukan perjalanan melalui seluruh aplikasi Anda. X-Ray mengumpulkan data yang dihasilkan oleh layanan individual seperti Amazon S3, AWS Lambda, dan Amazon EC2, serta banyak sumber daya yang membentuk aplikasi Anda. Ini memberi Anda dengan pandangan keseluruhan tentang bagaimana aplikasi Anda berkinerja.

Amazon S3 terintegrasi dengan X-Ray untuk untuk menyebarkan konteks pelacakan dan memberi Anda satu rantai permintaan dengan simpul upstream dan downstream. Jika layanan upstream menyertakan header pelacakan berformat valid dengan permintaan S3-nya, Amazon S3 meneruskan header pelacakan ketika mengirimkan pemberitahuan peristiwa ke layanan downstream seperti Lambda, Amazon SQS, dan Amazon SNS. Jika Anda memiliki semua layanan ini secara aktif terintegrasi dengan X-Ray, layanan tersebut terhubung dalam satu rantai permintaan untuk memberikan detail lengkap dari permintaan Amazon S3 Anda.

Untuk mengirim header pelacakan X-Ray melalui Amazon S3, Anda harus menyertakan X-Amzn-Trace-Id yang diformat dalam permintaan Anda. Anda juga dapat meng-instrumen klien Amazon S3 menggunakan SDK AWS X-Ray. Untuk daftar SDK yang didukung, lihat dokumentasi AWS X-Ray.

Peta layanan

Peta layanan X-Ray menunjukkan kepada Anda hubungan antara Amazon S3 dan layanan AWS lainnya serta sumber daya dalam aplikasi Anda dalam hampir secara langsung. Untuk melihat permintaan end-to-end menggunakan peta layanan X-Ray, Anda dapat menggunakan konsol X-Ray untuk melihat peta koneksi antara Amazon S3 dan layanan lain yang digunakan aplikasi Anda. Anda dapat dengan mudah mendeteksi di mana latensi tinggi terjadi, memvisualisasikan distribusi simpul untuk layanan ini, dan kemudian menelusuri layanan dan jalur tertentu yang memengaruhi performa aplikasi.

Analitik X-Ray

Anda juga dapat menggunakan konsol Analitik X-Ray untuk menganalisis pelacakan, melihat metrik seperti tingkat latensi dan kegagalan, serta menghasilkan wawasan untuk membantu Anda

mengidentifikasi dan memecahkan masalah. Konsol ini juga menampilkan metrik seperti tingkat latensi dan kegagalan rata-rata. Untuk informasi lebih lanjut, lihat konsol AWS X-Ray dalam Panduan Developer AWS X-Ray.

Wilayah yang Tersedia

AWS X-RayDukungan untuk Amazon S3 tersedia di semuaAWS X-RayKawasan. Untuk informasi lebih lanjut, lihat Amazon S3 dan AWS X-Ray dalam Panduan Developer AWS X-Ray.

Hosting situs web statis menggunakan Amazon S3

Anda dapat menggunakan Amazon S3 untuk host situs web statis. Pada situs web statis, halaman web individu termasuk konten statis. Mungkin juga berisi skrip sisi klien.

Sebaliknya, situs web dinamis mengandalkan pemrosesan sisi server, termasuk skrip sisi server seperti PHP, JSP, atau ASP.NET. Amazon S3 tidak mendukung skrip sisi server, tetapi AWS memiliki sumber daya lain untuk menghosting situs web dinamis. Untuk mempelajari lebih lanjut tentang hosting situs web AWS, lihat Web Hosting.

> **Note**
>
> Anda dapat menggunakan AWS Amplify Konsol untuk meng-host aplikasi web satu halaman. Konsol AWS Amplify mendukung aplikasi halaman tunggal yang dibuat dengan kerangka aplikasi halaman tunggal (misalnya, React JS, Vue JS, Angular JS, dan Nuxt) dan generator situs statis (misalnya, Gatsby JS, React-static, Jekyll, dan Hugo). Untuk informasi selengkapnya, lihat Memulai dalam Panduan Pengguna AWS Amplify .
> Titik akhir situs web Amazon S3 tidak mendukung HTTPS. Jika Anda ingin menggunakan HTTPS, Anda dapat menggunakan Amazon CloudFront untuk melayani situs web statis yang dihosting di Amazon S3. Untuk informasi selengkapnya, lihat Bagaimana cara menggunakan CloudFront untuk melayani permintaan HTTPS untuk bucket Amazon S3 saya? Untuk menggunakan HTTPS dengan domain khusus, lihat Mengonfigurasi situs web statis menggunakan domain khusus yang terdaftar di Route 53.

Untuk informasi selengkapnya tentang hosting situs web statis di Amazon S3, termasuk petunjuk dan step-by-step penelusuran, lihat topik berikut.

Topik

- Titik akhir situs web
- Mengaktifkan hosting situs web
- Mengonfigurasi dokumen indeks
- Mengonfigurasi dokumen kesalahan khusus
- Mengatur izin untuk akses situs web
- (Opsional) Mencatat lalu lintas web
- (Opsional) Mengonfigurasi pengalihan halaman web

Titik akhir situs web

Saat Anda mengonfigurasi bucket sebagai situs web statis, situs web ini tersedia di titik akhir situs web khusus Wilayah AWS. Titik akhir situs web berbeda dari titik akhir saat Anda mengirimkan permintaan API REST. Untuk informasi selengkapnya tentang perbedaan antara titik akhir, lihat Perbedaan utama antara titik akhir situs web dan titik akhir API REST.

Bergantung pada Wilayah Anda, titik akhir situs web Amazon S3 Anda mengikuti salah satu dari dua format ini.

- s3-situs web (-) Wilayah - `http://bucket-name.s3-website-Region.amazonaws.com`
- s3-situs web dot (.) Wilayah-`http://bucket-name.s3-website.Region.amazonaws.com`

URL ini mengembalikan dokumen indeks default yang Anda konfigurasikan untuk situs web. Untuk daftar lengkap titik akhir situs web Amazon S3, lihat Titik Akhir Situs Web Amazon S3.

> **Note**
>
> Untuk meningkatkan keamanan situs web statis Amazon S3 Anda, domain titik akhir situs web Amazon S3 (misalnya, s3 website-us-east - -1.amazonaws.com atau s3-website.ap-south-1.amazonaws.com) terdaftar di Daftar Akhiran Publik (PSL). Untuk keamanan lebih lanjut, kami menyarankan Anda menggunakan cookie dengan prefiks `__Host-` jika Anda perlu mengatur cookie sensitif di nama domain untuk situs web statis Amazon S3 Anda. Praktik ini akan membantu mempertahankan domain Anda dari upaya pemalsuan permintaan lintas situs (CSRF). Untuk informasi selengkapnya, lihat halaman Set-Cookie di Jaringan Pengembang Mozilla.

Jika Anda ingin situs web Anda menjadi publik, Anda harus membuat semua konten Anda dapat dibaca publik agar pelanggan Anda dapat mengaksesnya di titik akhir situs web. Untuk informasi selengkapnya, lihat Mengatur izin untuk akses situs web.

> **Important**
>
> Titik akhir situs web Amazon S3 tidak mendukung HTTPS atau titik akses. Jika Anda ingin menggunakan HTTPS, Anda dapat menggunakan Amazon CloudFront untuk melayani situs web statis yang dihosting di Amazon S3. Untuk informasi selengkapnya, lihat Bagaimana

cara menggunakan CloudFront untuk melayani permintaan HTTPS untuk bucket Amazon S3 saya? Untuk menggunakan HTTPS dengan domain khusus, lihat Mengonfigurasi situs web statis menggunakan domain khusus yang terdaftar di Route 53.
Bucket Peminta Membayar tidak mengizinkan akses melalui titik akhir situs web. Setiap permintaan ke bucket tersebut akan menerima respons 403 Akses Ditolak. Untuk informasi selengkapnya, lihat Menggunakan bucket Pembayaran Pemohon untuk transfer dan penggunaan penyimpanan.

Topik

- Contoh titik akhir situs web
- Menambahkan CNAME DNS
- Menggunakan domain khusus dengan Route 53
- Perbedaan utama antara titik akhir situs web dan titik akhir API REST

Contoh titik akhir situs web

Contoh berikut menunjukkan bagaimana Anda dapat mengakses bucket Amazon S3 yang dikonfigurasi sebagai situs web statis.

Example — Meminta objek di tingkat root

Untuk meminta objek tertentu yang disimpan di tingkat root dalam bucket, gunakan struktur URL berikut.

```
http://bucket-name.s3-website.Region.amazonaws.com/object-name
```

Misalnya, URL berikut meminta objek `photo.jpg` yang disimpan di tingkat root dalam bucket.

```
http://example-bucket.s3-website.us-west-2.amazonaws.com/photo.jpg
```

Example — Meminta satu objek dengan prefiks

Untuk meminta objek yang disimpan dalam folder di bucket Anda, gunakan struktur URL ini.

```
http://bucket-name.s3-website.Region.amazonaws.com/folder-name/object-name
```

URL berikut meminta objek `docs/doc1.html` tersebut dalam bucket Anda.

```
http://example-bucket.s3-website.us-west-2.amazonaws.com/docs/doc1.html
```

Menambahkan CNAME DNS

Jika Anda memiliki domain terdaftar, Anda dapat menambahkan entri CNAME DNS ke titik akhir situs web Amazon S3. Misalnya, jika Anda mendaftarkan domain `www.example-bucket.com`, Anda dapat membuat bucket `www.example-bucket.com`, dan menambahkan data CNAME DNS yang mengarah ke `www.example-bucket.com.s3-website.`*`Region`*`.amazonaws.com`. Semua permintaan untuk `http://www.example-bucket.com` dirutekan ke `www.example-bucket.com.s3-website.`*`Region`*`.amazonaws.com`.

Untuk informasi selengkapnya, lihat Menyesuaikan URL Amazon S3 dengan catatan CNAME.

Menggunakan domain khusus dengan Route 53

Daripada mengakses situs web menggunakan titik akhir situs web Amazon S3, Anda dapat menggunakan domain Anda sendiri yang terdaftar di Amazon Route 53 untuk menyajikan konten Anda—misalnya, `example.com`. Anda dapat menggunakan Amazon S3 dengan Route 53 untuk menghosting situs web di domain root. Misalnya, jika Anda memiliki domain root `example.com` dan Anda menghosting situs web Anda di Amazon S3, pengunjung situs web Anda dapat mengakses situs tersebut dari browser mereka dengan memasukkan salah satu `http://www.example.com` atau `http://example.com`.

Untuk panduan contoh, lihat Tutorial: Mengonfigurasi situs web statis menggunakan domain kustom yang terdaftar di Route 53.

Perbedaan utama antara titik akhir situs web dan titik akhir API REST

Titik akhir situs web Amazon S3 dioptimalkan untuk akses dari browser web. Tabel berikut merangkum perbedaan utama antara titik akhir API REST dan titik akhir situs web.

Perbedaan utama	Titik akhir API REST	Titik akhir situs web
Kontrol akses	Mendukung konten publik dan privat	Mendukung hanya konten yang dapat dibaca oleh publik

Perbedaan utama	Titik akhir API REST	Titik akhir situs web
Penanganan pesan kesalahan	Mengembalikan respons kesalahan yang diformat XML	Mengembalikan dokumen HTML
Dukungan pengalihan	Tidak berlaku	Mendukung pengalihan tingkat objek dan tingkat bucket
Permintaan yang didukung	Mendukung semua operasi bucket dan objek	Hanya mendukung permintaan GET dan HEAD pada objek
Tanggapan untuk permintaan GET dan HEAD di root bucket	Mengembalikan daftar kunci objek di dalam bucket	Mengembalikan dokumen indeks yang ditentukan dalam konfigurasi situs web
Dukungan Secure Sockets Layer (SSL)	Mendukung koneksi SSL	Tidak mendukung koneksi SSL

Untuk daftar lengkap titik akhir dan Wilayah Amazon S3, lihat Titik akhir dan kuota Amazon S3 di Referensi Umum AWS.

Mengaktifkan hosting situs web

Saat mengonfigurasi bucket sebagai situs web statis, Anda harus mengaktifkan hosting situs web statis, mengonfigurasi dokumen indeks, dan mengatur izin.

Anda dapat mengaktifkan hosting situs web statis menggunakan konsol Amazon S3, REST API, AWS SDK, atau. AWS CLI AWS CloudFormation

Untuk mengonfigurasi situs web Anda dengan domain khusus, lihat Tutorial: Mengonfigurasi situs web statis menggunakan domain kustom yang terdaftar di Route 53.

Menggunakan konsol S3

Untuk mengaktifkan hosting situs web statis

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang ingin Anda aktifkan hosting situs web statisnya.
3. Pilih Properti.
4. Di bagian bawah Hosting situs web statis, pilih Edit.
5. Pilih Gunakan bucket ini untuk menghosting situs web.
6. Di bagian bawah Hosting situs web statis, pilih Aktifkan.
7. Di Dokumen indeks, masukkan nama file dokumen indeks, biasanya `index.html`.

   Nama dokumen indeks peka huruf besar/kecil, dan harus sama persis dengan nama file dokumen indeks HTML yang ingin Anda unggah ke bucket S3 Anda. Saat Anda mengonfigurasi bucket untuk hosting situs web, Anda harus menentukan dokumen indeks. Amazon S3 mengembalikan dokumen indeks ini ketika permintaan dibuat ke domain root atau subfolder mana pun. Untuk informasi selengkapnya, lihat Mengonfigurasi dokumen indeks.
8. Agar dapat menyediakan dokumen kesalahan kustom Anda sendiri untuk kesalahan kelas 4XX, di Dokumen kesalahan, masukkan nama file dokumen kesalahan kustom.

   Nama dokumen kesalahan peka huruf besar/kecil, dan harus sama persis dengan nama file dokumen indeks HTML yang ingin Anda unggah ke bucket S3 Anda. Jika Anda tidak menentukan dokumen kesalahan khusus dan terjadi kesalahan, Amazon S3 mengembalikan dokumen kesalahan HTML default. Untuk informasi selengkapnya, lihat Mengonfigurasi dokumen kesalahan khusus.
9. (Opsional) Jika Anda ingin menentukan aturan pengalihan lanjutan, dalam Aturan pengalihan, masukkan JSON untuk menjelaskan aturannya.

   Misalnya, Anda dapat merutekan permintaan secara kondisional dengan nama kunci atau prefiks objek tertentu dalam permintaan tersebut. Untuk informasi selengkapnya, lihat Konfigurasikan aturan pengalihan untuk menggunakan pengalihan bersyarat lanjutan.
10. Pilih Simpan perubahan.

    Amazon S3 memungkinkan hosting situs web statis untuk bucket Anda. Di bagian bawah halaman, di bawah Hosting situs web statis, Anda melihat titik akhir situs web untuk bucket Anda.

11. Di bagian bawah Hosting situs web statis, perhatikan Titik Akhir.

    Titik Akhir adalah titik akhir situs web Amazon S3 untuk bucket Anda. Setelah Anda menyelesaikan konfigurasi bucket Anda sebagai situs web statis, Anda dapat menggunakan titik akhir ini untuk menguji situs web Anda.

Penggunaan API REST

Untuk informasi selengkapnya tentang mengirim permintaan REST secara langsung untuk mengaktifkan hosting situs web statis, lihat bagian berikut di Referensi API Amazon Simple Storage Service:

- PUT Bucket Situs web
- GET Bucket Situs web
- DELETE Bucket situs web

Menggunakan AWS SDK

Untuk meng-host situs web statis di Amazon S3, Anda mengonfigurasi bucket Amazon S3 untuk menghosting situs web lalu mengunggah konten situs web Anda ke bucket. Anda juga dapat menggunakan AWS SDK untuk membuat, memperbarui, dan menghapus konfigurasi situs web secara terprogram. SDK menyediakan kelas pembungkus di sekitar API REST Amazon S3. Jika aplikasi Anda memerlukannya, Anda dapat mengirim permintaan API REST langsung dari aplikasi Anda.

.NET

Contoh berikut menunjukkan cara menggunakan konfigurasi AWS SDK for .NET to manage website untuk bucket. Untuk menambahkan konfigurasi situs web ke bucket, berikan nama bucket dan konfigurasi situs web. Konfigurasi situs web harus menyertakan dokumen indeks dan dapat berisi dokumen kesalahan opsional. Dokumen ini harus disimpan dalam bucket. Untuk informasi selengkapnya, lihat Situs web PUT Bucket. Untuk informasi selengkapnya tentang fitur situs web Amazon S3, lihat Hosting situs web statis menggunakan Amazon S3.

Contoh kode C# berikut menambahkan konfigurasi situs web ke bucket tertentu. Konfigurasi menentukan dokumen indeks dan nama dokumen kesalahan. Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class WebsiteConfigTest
    {
        private const string bucketName = "*** bucket name ***";
        private const string indexDocumentSuffix = "*** index object key ***"; //
 For example, index.html.
        private const string errorDocument = "*** error object key ***"; // For
 example, error.html.
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;
        public static void Main()
        {
            client = new AmazonS3Client(bucketRegion);
            AddWebsiteConfigurationAsync(bucketName, indexDocumentSuffix,
 errorDocument).Wait();
        }

        static async Task AddWebsiteConfigurationAsync(string bucketName,
                                                       string indexDocumentSuffix,
                                                       string errorDocument)
        {
            try
            {
                // 1. Put the website configuration.
                PutBucketWebsiteRequest putRequest = new PutBucketWebsiteRequest()
                {
                    BucketName = bucketName,
                    WebsiteConfiguration = new WebsiteConfiguration()
                    {
                        IndexDocumentSuffix = indexDocumentSuffix,
                        ErrorDocument = errorDocument
                    }
                };
```

```
                PutBucketWebsiteResponse response = await
client.PutBucketWebsiteAsync(putRequest);

                // 2. Get the website configuration.
                GetBucketWebsiteRequest getRequest = new GetBucketWebsiteRequest()
                {
                    BucketName = bucketName
                };
                GetBucketWebsiteResponse getResponse = await
client.GetBucketWebsiteAsync(getRequest);
                Console.WriteLine("Index document: {0}",
getResponse.WebsiteConfiguration.IndexDocumentSuffix);
                Console.WriteLine("Error document: {0}",
getResponse.WebsiteConfiguration.ErrorDocument);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
writing an object", e.Message);
            }
        }
    }
}
```

PHP

Contoh PHP berikut ini menambahkan konfigurasi situs web ke bucket yang ditentukan. Metode `create_website_config` secara eksplisit menyediakan nama dokumen indeks dan dokumen kesalahan. Contoh ini juga mengambil konfigurasi situs web dan mencetak respons. Untuk informasi selengkapnya tentang fitur situs web Amazon S3, lihat Hosting situs web statis menggunakan Amazon S3.

Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

```
 require 'vendor/autoload.php';

use Aws\S3\S3Client;
```

```
$bucket = '*** Your Bucket Name ***';

$s3 = new S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);


// Add the website configuration.
$s3->putBucketWebsite([
    'Bucket'                => $bucket,
    'WebsiteConfiguration'  => [
        'IndexDocument' => ['Suffix' => 'index.html'],
        'ErrorDocument' => ['Key' => 'error.html']
    ]
]);

// Retrieve the website configuration.
$result = $s3->getBucketWebsite([
    'Bucket' => $bucket
]);
echo $result->getPath('IndexDocument/Suffix');

// Delete the website configuration.
$s3->deleteBucketWebsite([
    'Bucket' => $bucket
]);
```

Menggunakan AWS CLI

Untuk informasi selengkapnya tentang penggunaan AWS CLI untuk mengonfigurasi bucket S3 sebagai situs web statis, lihat situs web di Referensi AWS CLI Perintah.

Selanjutnya, Anda harus mengonfigurasi dokumen indeks dan mengatur izin. Untuk informasi selengkapnya, lihat Mengonfigurasi dokumen indeks dan Mengatur izin untuk akses situs web.

Anda juga dapat secara opsional mengonfigurasi dokumen kesalahan, catatan lalu lintas web, atau arahkan ulang.

Mengonfigurasi dokumen indeks

Saat Anda mengaktifkan hosting situs web, Anda juga harus mengonfigurasi dan mengunggah dokumen indeks. Dokumen indeks adalah halaman web yang dikembalikan Amazon S3 ketika permintaan dibuat ke root situs web atau setiap subfolder. Misalnya, jika pengguna memasukkan `http://www.example.com` di browser, pengguna tidak meminta halaman tertentu. Dalam hal ini, Amazon S3 menyajikan dokumen indeks, yang kadang-kadang disebut sebagai halaman default.

Saat mengaktifkan hosting situs web statis untuk bucket Anda, Anda memasukkan nama dokumen indeks (misalnya, `index.html`). Setelah Anda mengaktifkan hosting situs web statis untuk bucket, Anda mengunggah file HTML dengan nama dokumen indeks ke bucket Anda.

Garis miring di akhir URL tingkat root bersifat opsional. Misalnya, jika Anda mengonfigurasi situs web Anda dengan `index.html` sebagai dokumen indeks, salah satu dari URL berikut akan mengembalikan `index.html`.

```
http://example-bucket.s3-website.Region.amazonaws.com/
http://example-bucket.s3-website.Region.amazonaws.com
```

Untuk informasi selengkapnya tentang titik akhir situs web Amazon S3, lihat Titik akhir situs web.

Indeks dokumen dan folder

Di Amazon S3, bucket adalah kontainer datar objek. Bucket tidak menyediakan penyusunan hierarkis seperti halnya sistem file di komputer Anda. Namun, Anda dapat membuat hierarki logis dengan menggunakan nama kunci objek yang menyiratkan struktur folder.

Sebagai contoh, anggaplah bucket dengan tiga objek yang memiliki nama utama berikut. Meskipun ketiga objek ini disimpan tanpa organisasi hierarki fisik, Anda dapat menyimpulkan struktur folder logis berikut dari nama kunci:

- `sample1.jpg`—Objek berada di root bucket.
- `photos/2006/Jan/sample2.jpg`—Objek berada di subfolder `photos/2006/Jan`.
- `photos/2006/Feb/sample3.jpg`—Objek berada di subfolder `photos/2006/Feb`.

Di konsol Amazon S3, Anda juga dapat membuat folder di bucket. Misalnya, Anda dapat membuat folder dengan nama `photos`. Anda dapat mengunggah objek ke bucket atau ke folder `photos` di dalam bucket. Jika Anda menambahkan objek `sample.jpg` ke bucket, nama kunci adalah

`sample.jpg`. Jika Anda meng-upload objek ke folder `photos`, nama kunci objek adalah `photos/sample.jpg`.

Jika Anda membuat struktur folder dalam bucket, Anda harus memiliki dokumen indeks di setiap tingkat. Dalam setiap folder, dokumen indeks harus mempunyai nama yang sama, sebagai contoh, `index.html`. Saat pengguna menentukan URL yang menyerupai pencarian folder, ada atau tidak adanya garis miring menentukan perilaku situs web. Misalnya, URL berikut, dengan garis miring, mengembalikan dokumen indeks `photos/index.html`.

```
http://bucket-name.s3-website.Region.amazonaws.com/photos/
```

Namun, jika Anda mengecualikan garis miring dari URL sebelumnya, Amazon S3 akan terlebih dahulu mencari objek `photos` dalam bucket. Jika objek `photos` tidak ditemukan, maka Amazon S3 akan mencari dokumen indeks, `photos/index.html`. Jika dokumen itu ditemukan, Amazon S3 mengembalikan pesan `302 Found` dan menunjuk ke kunci `photos/`. Untuk permintaan berikutnya ke `photos/`, Amazon S3 mengembalikan `photos/index.html`. Jika dokumen indeks tidak ditemukan, Amazon S3 mengembalikan kesalahan.

Mengonfigurasi dokumen indeks

Untuk mengonfigurasi dokumen indeks menggunakan konsol S3, gunakan prosedur berikut. Anda juga dapat mengonfigurasi dokumen indeks menggunakan REST API, AWS SDK AWS CLI, atau AWS CloudFormation.

> **Note**
>
> Dalam bucket berkemampuan Penentuan Versi, Anda dapat mengunggah beberapa salinan `index.html`, tetapi hanya versi terbaru yang akan diselesaikan. Untuk informasi selengkapnya tentang Penentuan Versi S3, lihat Menggunakan Penentuan Versi dalam bucket S3.

Saat mengaktifkan hosting situs web statis untuk bucket Anda, Anda memasukkan nama dokumen indeks (misalnya, **`index.html`**). Setelah Anda mengaktifkan hosting situs web statis untuk bucket, Anda mengunggah file HTML dengan nama dokumen indeks ke bucket Anda.

Untuk mengonfigurasi dokumen indeks

1. Buat file `index.html`.

Jika Anda tidak memiliki file `index.html`, Anda dapat menggunakan HTML berikut ini untuk membuatnya:

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
    <title>My Website Home Page</title>
</head>
<body>
  <h1>Welcome to my website</h1>
  <p>Now hosted on Amazon S3!</p>
</body>
</html>
```

2. Simpan file indeks secara lokal.

   Nama file dokumen indeks harus sama persis dengan nama dokumen indeks yang Anda masukkan ke dalam kotak dialog Hosting situs web statis. Nama dokumen indeks peka huruf besar/kecil. Misalnya, jika Anda memasukkan `index.html` untuk Dokumen indeks dalam kotak dialog Hosting situs web statis, nama file dokumen indeks Anda juga harus dan bukan `Index.html`.

3. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
4. Di daftar Bucket, pilih nama bucket yang ingin Anda gunakan untuk meng-host situs web statis.
5. Aktifkan hosting situs web statis untuk bucket Anda, lalu masukkan nama persis dokumen indeks Anda (misalnya, `index.html`). Untuk informasi selengkapnya, lihat Mengaktifkan hosting situs web.

   Setelah mengaktifkan hosting situs web statis, lanjutkan ke langkah 6.

6. Untuk mengunggah dokumen indeks ke bucket Anda, lakukan salah satu hal berikut ini:

   - Seret dan jatuhkan file indeks ke dalam daftar bucket konsol.
   - Pilih Unggah, dan ikuti petunjuk untuk memilih dan mengunggah file indeks.

   Untuk step-by-step instruksi, lihatMengunggah Objek.

7. (Opsional) Unggah konten situs web lain ke bucket Anda.

Selanjutnya, Anda harus mengatur izin akses situs web. Untuk informasi, lihat Mengatur izin untuk akses situs web.

Anda juga dapat secara opsional mengonfigurasi dokumen kesalahan, catatan lalu lintas web, atau arahkan ulang.

Mengonfigurasi dokumen kesalahan khusus

Setelah Anda mengonfigurasi bucket Anda sebagai situs web statis, ketika kesalahan terjadi, Amazon S3 mengembalikan dokumen kesalahan HTML. Anda dapat secara opsional mengonfigurasi bucket Anda dengan dokumen kesalahan kustom sehingga Amazon S3 mengembalikan dokumen saat kesalahan terjadi.

> **Note**
>
> Beberapa browser menampilkan pesan kesalahan mereka sendiri saat terjadi kesalahan, mengabaikan dokumen kesalahan yang dikembalikan Amazon S3. Misalnya, ketika terjadi kesalahan HTTP 404 Not Found, Google Chrome mungkin mengabaikan dokumen kesalahan yang dikembalikan Amazon S3 dan menampilkan kesalahannya sendiri.

Topik

- Kode respons HTTP Amazon S3
- Mengonfigurasi dokumen kesalahan khusus

Kode respons HTTP Amazon S3

Tabel berikut mencantumkan subset kode respons HTTP yang dikembalikan Amazon S3 saat kesalahan terjadi.

Kode kesalahan HTTP	Deskripsi
301 Dipindahkan Secara Permanen	Saat pengguna mengirimkan permintaan secara langsung ke titik akhir situs web Amazon S3 (`http://s3-website.` *`Region`*`.amazonaws.com/`), Amazon S3 mengembalikan 301 Dipindahkan Secara Permanen menanggap

Kode kesalahan HTTP	Deskripsi
	i dan mengalihkan permintaan tersebut ke `https://aws.amazon.com/s3/`.
302 Ditemukan	Ketika Amazon S3 menerima permintaan untuk kunci `x`, `http://`*`bucket-name`*`.s3-website.`*`Region`*`.amazonaws.com/x`, tanpa garis miring, pertama-tama Amazon akan mencari objek dengan nama kunci `x`. Jika objek tidak ditemukan, Amazon S3 menentukan bahwa permintaan itu untuk subfolder `x` dan mengalihkan permintaan dengan menambahkan garis miring di akhir, dan mengembalikan 302 Ditemukan.
304 Tidak Dimodifikasi	Amazon S3 menggunakan header permintaan `If-Modified-Since`, `If-Unmodified-Since`, `If-Match` dan/atau `If-None-Match` untuk menentukan apakah objek yang diminta sama dengan salinan yang disimpan di cache yang dipegang oleh klien. Jika objek sama, titik akhir situs web mengembalikan 304 Tidak Dimodifikasi tanggapan mereka.
400 Permintaan Berformat Salah	Titik akhir situs web merespons dengan 400 Permintaan Berformat Salah saat pengguna mencoba mengakses bucket melalui titik akhir regional yang salah.
403 Dilarang	Titik akhir situs web merespons dengan 403 Dilarang ketika permintaan pengguna menerjemahkan ke objek yang tidak dapat dibaca oleh publik. Pemilik objek harus membuat objek dapat dibaca oleh publik menggunakan kebijakan bucket atau ACL.

Kode kesalahan HTTP	Deskripsi
404 Tidak Ditemukan	Titik akhir situs web merespons dengan 404 Tidak Ditemukan karena alasan berikut: • Amazon S3 menentukan bahwa URL situs web mengacu pada kunci objek yang tidak ada. • Amazon S3 menyimpulkan bahwa permintaan tersebut adalah untuk dokumen indeks yang tidak ada. • Bucket yang ditentukan dalam URL tidak tersedia. • Terdapat bucket yang disebutkan dalam URL, tetapi tidak dikonfigurasi sebagai situs web. Anda dapat membuat dokumen kustom yang dikembalikan untuk 404 Tidak Ditemukan. Pastikan dokumen diunggah ke bucket yang dikonfigurasi sebagai situs web, dan bahwa konfigurasi hosting situs web diatur untuk menggunakan dokumen. Untuk informasi tentang cara Amazon S3 menginterpretasikan URL sebagai permintaan objek atau dokumen indeks, lihat Mengonfigurasi dokumen indeks.
500 Kesalahan Layanan	Titik akhir situs web merespons dengan 500 Kesalahan Layanan saat terjadi kesalahan server internal.
503 Layanan Tidak Tersedia	Titik akhir situs web merespons dengan 503 Layanan Tidak Tersedia ketika Amazon S3 menentukan bahwa Anda perlu mengurangi tingkat permintaan Anda.

Untuk setiap kesalahan ini, Amazon S3 mengembalikan pesan HTML yang sudah ditentukan sebelumnya. Berikut ini adalah contoh pesan HTML yang dikembalikan untuk respons 403 Dilarang.

403 Forbidden

- Code: AccessDenied
- Message: Access Denied
- RequestId: 873CA367A51F7EC7
- HostId: DdQezl9vkuw5luD5HKsFaTDm9KH4PZzCPRkW3igimILbTu1DiYlvXjgyd7pVxq32

An Error Occurred While Attempting to Retrieve a Custom Error Document

- Code: AccessDenied
- Message: Access Denied

Mengonfigurasi dokumen kesalahan khusus

Saat Anda mengonfigurasi bucket sebagai situs web statis, Anda dapat memberikan dokumen kesalahan khusus yang berisi pesan kesalahan dan bantuan tambahan yang mudah digunakan. Amazon S3 mengembalikan dokumen kesalahan kustom Anda hanya untuk kode kesalahan kelas HTTP 4XX saja.

Untuk mengonfigurasi dokumen kesalahan kustom menggunakan konsol S3, ikuti langkah-langkah di bawah ini. Anda juga dapat mengonfigurasi dokumen kesalahan menggunakan REST API, AWS SDK AWS CLI, atau AWS CloudFormation. Untuk informasi selengkapnya, lihat berikut ini:

- PutBucketWebsitedi Referensi API Layanan Penyimpanan Sederhana Amazon
- AWS::S3::Bucket WebsiteConfiguration di Panduan Pengguna AWS CloudFormation
- put-bucket-websitedalam Referensi AWS CLI Perintah

Saat mengaktifkan hosting situs web statis untuk bucket Anda, Anda memasukkan nama dokumen kesalahan (misalnya, **`404.html`**). Setelah Anda mengaktifkan hosting situs web statis untuk bucket, Anda mengunggah file HTML dengan nama dokumen indeks ke bucket Anda.

Untuk mengonfigurasi dokumen kesalahan

1. Membuat dokumen kesalahan, misalnya `404.html`.

2. Simpan file dokumen kesalahan secara lokal.

   Nama dokumen kesalahan peka huruf besar/kecil, dan harus sama persis dengan nama yang Anda masukkan saat Anda mengaktifkan hosting situs web statis. Misalnya, jika Anda memasukkan `404.html` sebagai nama Dokumen kesalahan di kotak dialog Hosting situs web statis, nama file dokumen kesalahan Anda juga harus bernama `404.html`.

3. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
4. Di daftar Bucket, pilih nama bucket yang ingin Anda gunakan untuk meng-host situs web statis.
5. Aktifkan hosting situs web statis untuk bucket Anda, lalu masukkan nama persis dokumen indeks Anda (misalnya, `404.html`). Untuk informasi selengkapnya, lihat Mengaktifkan hosting situs web dan Mengonfigurasi dokumen kesalahan khusus.

   Setelah mengaktifkan hosting situs web statis, lanjutkan ke langkah 6.

6. Untuk mengunggah dokumen kesalahan ke bucket Anda, lakukan salah satu hal berikut ini:

   - Seret dan jatuhkan file dokumen kesalahan ke dalam daftar bucket konsol.
   - Pilih Unggah, dan ikuti petunjuk untuk memilih dan mengunggah file indeks.

   Untuk step-by-step instruksi, lihatMengunggah Objek.

Mengatur izin untuk akses situs web

Saat Anda mengonfigurasikan bucket sebagai situs web statis, jika Anda ingin situs web Anda bersifat publik, Anda dapat memberikan akses baca publik. Agar bucket dapat dibaca publik, Anda harus menonaktifkan pengaturan akses publik untuk bucket dan menulis kebijakan bucket yang memberi akses baca publik. Jika bucket Anda berisi objek yang tidak dimiliki oleh pemilik bucket, Anda mungkin juga perlu menambahkan daftar kontrol akses (ACL) objek yang memberi setiap orang akses baca.

Jika Anda tidak ingin menonaktifkan blokir pengaturan akses publik untuk bucket Anda tetapi Anda masih ingin situs web Anda menjadi publik, Anda dapat membuat CloudFront distribusi Amazon untuk melayani situs web statis Anda. Untuk informasi selengkapnya, lihat Mempercepat situs web Anda dengan Amazon CloudFront atau Menggunakan CloudFront distribusi Amazon untuk menayangkan situs web statis di Panduan Pengembang Amazon Route 53.

> **Note**
>
> Di titik akhir situs web, jika pengguna meminta objek yang tidak ada, Amazon S3 mengembalikan kode respons HTTP `404 (Not Found)`. Jika objek ada tetapi Anda belum memberikan izin baca padanya, titik akhir situs web mengembalikan kode respons HTTP `403 (Access Denied)`. Pengguna dapat menggunakan kode respons untuk menyimpulkan apakah objek tertentu ada. Jika Anda tidak menginginkan perilaku ini, Anda tidak boleh mengaktifkan dukungan situs web untuk bucket Anda.

Topik

- Langkat 1: Edit pengaturan Blokir Akses Publik S3
- Langkah 2: Menambahkan kebijakan bucket
- Daftar kontrol akses objek

Langkat 1: Edit pengaturan Blokir Akses Publik S3

Jika Anda ingin mengonfigurasi bucket yang ada sebagai situs web statis yang memiliki akses publik, Anda harus mengedit pengaturan Blokir Akses Publik untuk bucket tersebut. Anda mungkin juga perlu mengedit pengaturan Blokir Akses Publik tingkat akun Anda. Amazon S3 menerapkan kombinasi paling ketat dari pengaturan akses publik tingkat bucket dan tingkat akun.

Misalnya, jika Anda mengizinkan akses publik untuk bucket tetapi memblokir semua akses publik di tingkat akun, Amazon S3 akan terus memblokir akses publik ke bucket. Dalam skenario ini, Anda harus mengedit pengaturan Blokir Akses Publik tingkat bucket dan tingkat akun Anda. Untuk informasi selengkapnya, lihat Melakukan blok akses publik ke penyimpanan Amazon S3 Anda.

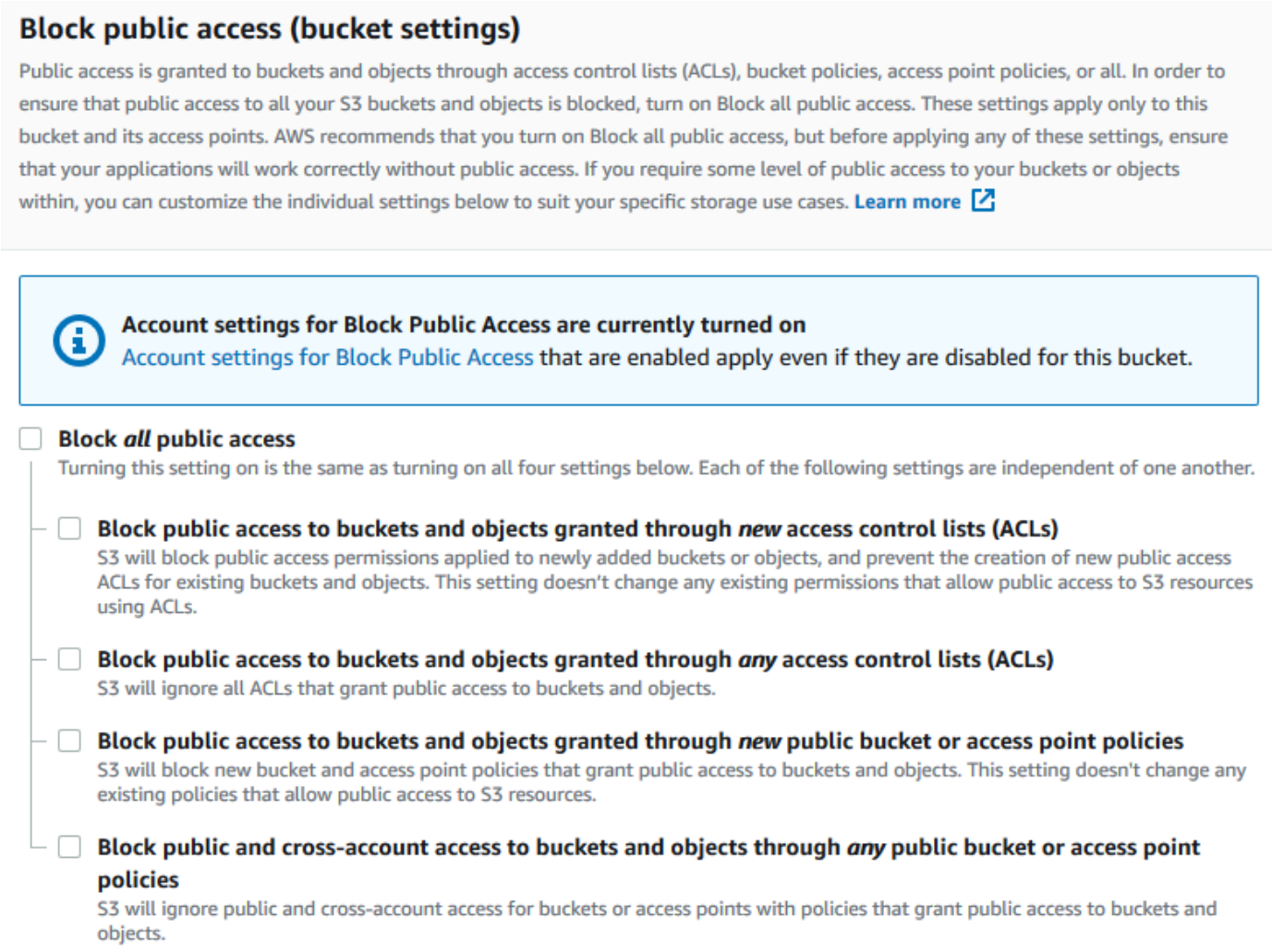
Secara default, Amazon S3 memblokir akses publik ke akun dan bucket Anda. Jika Anda ingin menggunakan bucket untuk menghosting situs web statis, Anda dapat menggunakan langkah-langkah ini untuk mengedit pengaturan blokir akses publik Anda.

> **Warning**
>
> Sebelum Anda menyelesaikan langkah ini, tinjau Melakukan blok akses publik ke penyimpanan Amazon S3 Anda untuk memastikan bahwa Anda telah memahami dan menerima risiko yang terkait dengan mengizinkan akses publik. Saat Anda mematikan pengaturan blokir akses publik untuk membuat bucket Anda menjadi publik, siapa pun di

internet dapat mengakses bucket Anda. Kami sarankan agar Anda memblokir semua akses publik ke bucket Anda.

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Pilih nama bucket yang telah Anda konfigurasikan sebagai situs web statis.
3. Pilih Izin.
4. Di bagian bawah Blokir akses publik (pengaturan bucket), pilih Edit.
5. Kosongkan Blokir semua akses publik, lalu pilih Simpan perubahan.

> **Warning**
>
> Sebelum Anda menyelesaikan langkah ini, tinjau Melakukan blok akses publik ke penyimpanan Amazon S3 Anda untuk memastikan bahwa Anda telah memahami dan menerima risiko yang terkait dengan mengizinkan akses publik. Saat Anda mematikan pengaturan blokir akses publik untuk membuat bucket Anda menjadi publik, siapa pun di internet dapat mengakses bucket Anda. Kami sarankan agar Anda memblokir semua akses publik ke bucket Anda.



Amazon S3 menonaktifkan pengaturan Blokir Akses Publik untuk bucket Anda. Untuk membuat situs web publik statis, Anda mungkin harus mengedit pengaturan Blokir Akses Publik untuk akun Anda sebelum menambahkan kebijakan bucket. Jika pengaturan akun untuk Blokir Akses Publik saat ini diaktifkan, Anda akan melihat catatan di Blokir akses publik (pengaturan bucket).

Langkah 2: Menambahkan kebijakan bucket

Untuk membuat objek dalam bucket Anda dapat dibaca publik, Anda harus menulis kebijakan bucket yang memberikan setiap orang izin `s3:GetObject`.

Setelah Anda mengedit pengaturan Blokir Akses Publik S3, Anda dapat menambahkan kebijakan bucket untuk memberikan akses baca publik ke bucket Anda. Saat Anda memberikan akses baca publik, siapa pun di internet dapat mengakses bucket Anda.

> **Important**
>
> Kebijakan berikut ini hanya merupakan contoh, dan memungkinkan akses penuh ke konten bucket Anda. Sebelum melanjutkan langkah ini, tinjau Bagaimana saya dapat mengamankan file dalam bucket Amazon S3 saya? untuk memastikan bahwa Anda telah memahami praktik terbaik untuk mengamankan file dalam bucket S3, dan risiko yang terlibat dalam pemberian akses publik.

1. Di bagian bawah Bucket, pilih nama bucket Anda.
2. Pilih Izin.
3. Di Bawah Kebijakan bucket, pilih Edit.
4. Untuk memberikan akses baca bagi publik untuk situs web Anda, salin kebijakan kelompok berikut, dan tempelkan di Editor kebijakan bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::Bucket-Name/*"
            ]
        }
    ]
}
```

5. Perbarui `Resource` dengan nama bucket Anda.

   Dalam contoh kebijakan bucket sebelumnya, *`Nama-Bucket`* adalah placeholder untuk nama bucket tersebut. Untuk menggunakan kebijakan bucket ini dengan bucket Anda sendiri, Anda harus memperbarui nama ini agar sesuai dengan nama bucket Anda.
6. Pilih Simpan perubahan.

Pesan akan muncul, yang menunjukkan bahwa kebijakan bucket telah berhasil ditambahkan.

Jika Anda melihat kesalahan yang mengatakan `Policy has invalid resource`, konfirmasikan bahwa nama bucket dalam kebijakan bucket tersebut sesuai dengan nama bucket Anda. Untuk informasi tentang menambahkan kebijakan bucket, lihat Bagaimana cara menambahkan kebijakan S3 bucket?

Jika Anda mendapatkan pesan kesalahan dan tidak dapat menyimpan kebijakan bucket, periksa pengaturan akun dan bucket Blokir Akses Publik untuk mengonfirmasi bahwa Anda mengizinkan akses publik ke bucket.

Daftar kontrol akses objek

Anda dapat menggunakan kebijakan bucket untuk memberikan izin baca publik kepada objek Anda. Namun, kebijakan bucket hanya berlaku untuk objek yang dimiliki oleh pemilik bucket. Apabila bucket Anda berisi objek yang bukan milik pemilik bucket, pemilik bucket harus menggunakan daftar kontrol akses (ACL) objek untuk memberikan izin BACA publik pada objek tersebut.

Kepemilikan Objek S3 adalah pengaturan tingkat bucket Amazon S3 yang dapat Anda gunakan untuk mengontrol kepemilikan objek yang diunggah ke bucket Anda, serta menonaktifkan atau mengaktifkan ACL. Secara default, Kepemilikan Objek diatur ke pengaturan yang diberlakukan pemilik Bucket, dan semua ACL dinonaktifkan. Ketika ACL dinonaktifkan, pemilik bucket memiliki semua objek di bucket dan mengelola akses ke objek-objek tersebut secara eksklusif dengan menggunakan kebijakan manajemen akses.

Mayoritas kasus penggunaan modern di Amazon S3 tidak lagi memerlukan penggunaan ACL. Kami menyarankan agar ACL dinonaktifkan, kecuali dalam keadaan yang tidak biasa ketika Anda perlu mengontrol akses untuk setiap objek secara individual. Dengan ACL dinonaktifkan, Anda dapat menggunakan kebijakan untuk mengontrol akses ke semua objek di bucket, terlepas dari siapa yang mengunggah objek ke bucket Anda. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.

> **Important**
>
> Jika bucket Anda menggunakan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek S3, Anda harus menggunakan kebijakan untuk memberikan akses ke bucket Anda, serta objek di dalamnya. Dengan mengaktifkan pengaturan yang diberlakukan pemilik Bucket, permintaan untuk mengatur daftar kontrol akses

(ACL) atau memperbarui ACL akan gagal dan mengembalikan kode kesalahan `AccessControlListNotSupported`. Permintaan untuk membaca ACL masih didukung.

Untuk membuat objek yang dapat dibaca oleh publik dengan menggunakan ACL, berikan izin READ kepada kelompok `AllUsers`, sebagaimana ditunjukkan dalam elemen pemberian berikut. Tambahkan elemen pemberian ini ke objek ACL. Untuk informasi tentang pengelolaan ACL, lihat Gambaran umum daftar kontrol akses (ACL).

```
<Grant>
  <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:type="Group">
    <URI>http://acs.amazonaws.com/groups/global/AllUsers</URI>
  </Grantee>
  <Permission>READ</Permission>
</Grant>
```

(Opsional) Mencatat lalu lintas web

Anda dapat secara opsional mengaktifkan pencatatan akses server Amazon S3 untuk bucket yang dikonfigurasi sebagai situs web statis. Pencatatan akses server menyediakan catatan terperinci untuk permintaan yang dilakukan ke bucket Anda. Untuk informasi selengkapnya, lihat Pencatatan permintaan dengan pencatatan akses server. Jika Anda berencana menggunakan Amazon CloudFront untuk mempercepat situs web Anda, Anda juga dapat menggunakan CloudFront logging. Untuk informasi selengkapnya, lihat Mengonfigurasi dan Menggunakan Log Akses di Panduan CloudFront Pengembang Amazon.

Untuk mengaktifkan log akses server untuk bucket situs web statis Anda

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Dalam Wilayah yang sama tempat Anda membuat bucket yang dikonfigurasi sebagai situs web statis, buat bucket untuk log, misalnya `logs.example.com`.
3. Buat folder untuk file log pencatatan akses server (misalnya, `logs`).
4. (Opsional) Jika Anda ingin menggunakan CloudFront untuk meningkatkan kinerja situs web Anda, buat folder untuk file CloudFront log (misalnya,`cdn`).

   Untuk informasi selengkapnya, lihat Mempercepat situs web Anda dengan Amazon CloudFront.
5. Di daftar Bucket pilih bucket Anda.

6. Pilih Properti.
7. Di bagian bawah Pencatatan akses server, pilih Edit.
8. Pilih Aktifkan.
9. Under the Target bucket, choose the bucket and folder destination for the server access:
   - Jelajahi ke lokasi folder dan bucket:
     1. Pilih Jelajahi S3.
     2. Pilih nama bucket, lalu pilih folder log.
     3. Pilih Pilih jalur.
   - Masukkan alur bucket S3, misalnya, **`s3://logs.example.com/logs/`**.
10. Pilih Simpan perubahan.

    Dalam bucket log, sekarang dapat mengakses log Anda. Amazon S3 menulis log akses situs web ke bucket log Anda setiap 2 jam.

(Opsional) Mengonfigurasi pengalihan halaman web

Jika bucket Amazon S3 Anda dikonfigurasi untuk hosting situs web statis, Anda dapat mengonfigurasi pengalihan untuk bucket atau objek Anda di dalamnya. Anda memiliki opsi berikut untuk mengonfigurasikan pengalihan.

Topik

- Permintaan pengalihan untuk titik akhir situs web bucket Anda ke bucket atau domain lain
- Konfigurasikan aturan pengalihan untuk menggunakan pengalihan bersyarat lanjutan
- Mengalihkan permintaan untuk objek

Permintaan pengalihan untuk titik akhir situs web bucket Anda ke bucket atau domain lain

Anda dapat mengalihkan semua permintaan ke sebuah titik akhir situs web untuk sebuah bucket ke bucket atau domain yang lain. Jika Anda mengalihkan semua permintaan, permintaan apapun yang dibuat ke titik akhir situs web dialihkan ke bucket atau domain tertentu.

Misalnya, jika domain root Anda adalah `example.com`, dan Anda ingin melayani permintaan untuk keduanya `http://example.com` dan `http://www.example.com`, Anda harus membuat dua

bucket bernama `example.com` dan `www.example.com`. Kemudian, pertahankan konten di bucket `example.com`, dan konfigurasikan bucket `www.example.com` lainnya untuk mengalihkan semua permintaan ke bucket `example.com`. Untuk informasi selengkapnya, lihat Mengonfigurasi Situs Web Statis Menggunakan Nama Domain Khusus.

Untuk mengalihkan permintaan titik akhir situs web bucket

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di Bawah Bucket, pilih nama bucket tempat Anda ingin mengalihkan permintaan (misalnya, `www.example.com`).
3. Pilih Properti.
4. Di bagian bawah Hosting situs web statis, pilih Edit.
5. Pilih Alihkan permintaan objek.
6. Di kotak Nama host, masukkan titik akhir situs web untuk bucket atau domain khusus Anda.

   Misalnya, jika Anda mengalihkan ke alamat domain root, Anda akan memasukkan **`example.com`**.
7. Untuk Protokol, pilih protokol untuk permintaan pengalihan (tidak ada,http, atau https).

   Jika Anda tidak menentukan protokol, opsi default adalah tidak ada.
8. Pilih Simpan perubahan.

Konfigurasikan aturan pengalihan untuk menggunakan pengalihan bersyarat lanjutan

Menggunakan aturan pengalihan lanjutan, Anda dapat mengirimkan permintaan secara bersyarat sesuai dengan nama kunci, prefiks tertentu dalam permintaan, atau kode tanggapan. Misalnya, bayangkan bahwa Anda menghapus atau mengganti nama sebuah objek di dalam bucket Anda. Anda dapat menambahkan aturan perutean yang mengarahkan permintaan ke objek lain. Jika Anda ingin membuat folder menjadi tidak tersedia, Anda dapat menambahkan aturan perutean untuk mengalihkan permintaan ke halaman web lain. Anda juga dapat menambahkan aturan perutean untuk menangani kondisi kesalahan dengan mengirimkan permintaan yang mengembalikan kesalahan ke domain lain saat kesalahan diproses.

Saat mengaktifkan hosting situs web statis untuk bucket Anda, Anda dapat menentukan aturan pengalihan lanjutan secara opsional. Amazon S3 memiliki batasan 50 aturan perutean per konfigurasi

situs web. Jika Anda memerlukan lebih dari 50 aturan perutean, Anda dapat menggunakan pengalihan objek. Untuk informasi selengkapnya, lihat Menggunakan konsol S3.

Untuk informasi selengkapnya tentang mengonfigurasi aturan perutean menggunakan REST API, lihat PutBucketWebsitedi Referensi API Amazon Simple Storage Service.

> **Important**
>
> Untuk membuat aturan pengalihan di konsol Amazon S3, Anda harus menggunakan JSON. Untuk contoh JSON, lihat Contoh aturan pengalihan.

Untuk mengonfigurasi aturan pengalihan situs web statis

Untuk menambahkan aturan pengalihan ke bucket yang telah mengaktifkan hosting situs web statis, ikuti langkah-langkah ini.

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang telah Anda konfigurasikan sebagai situs web statis.
3. Pilih Properti.
4. Di bagian bawah Hosting situs web statis, pilih Edit.
5. Di kotak Aturan pengalihan, masukkan aturan pengalihan Anda di JSON.

   Pada konsol S3 Anda mendeskripsikan aturan menggunakan JSON. Untuk contoh JSON, lihat Contoh aturan pengalihan. Amazon S3 memiliki batasan 50 aturan perutean per konfigurasi situs web.
6. Pilih Simpan perubahan.

Elemen aturan perutean

Berikut ini adalah sintaks umum untuk menentukan aturan perutean dalam konfigurasi situs web di JSON dan XML Untuk mengonfigurasi aturan pengalihan di konsol S3 yang baru, Anda harus menggunakan JSON. Untuk contoh JSON, lihat Contoh aturan pengalihan.

JSON

```
[
    {
```

```
      "Condition": {
        "HttpErrorCodeReturnedEquals": "string",
        "KeyPrefixEquals": "string"
      },
      "Redirect": {
        "HostName": "string",
        "HttpRedirectCode": "string",
        "Protocol": "http"|"https",
        "ReplaceKeyPrefixWith": "string",
        "ReplaceKeyWith": "string"
      }
    }
  ]

Note: Redirect must each have at least one child element. You can have either
 ReplaceKeyPrefix with or ReplaceKeyWith but not both.
```

XML

```
<RoutingRules> =
    <RoutingRules>
         <RoutingRule>...</RoutingRule>
         [<RoutingRule>...</RoutingRule>
         ...]
    </RoutingRules>

<RoutingRule> =
   <RoutingRule>
      [ <Condition>...</Condition> ]
      <Redirect>...</Redirect>
   </RoutingRule>

<Condition> =
   <Condition>
      [ <KeyPrefixEquals>...</KeyPrefixEquals> ]
      [ <HttpErrorCodeReturnedEquals>...</HttpErrorCodeReturnedEquals> ]
   </Condition>
    Note: <Condition> must have at least one child element.

<Redirect> =
   <Redirect>
      [ <HostName>...</HostName> ]
      [ <Protocol>...</Protocol> ]
```

```
      [ <ReplaceKeyPrefixWith>...</ReplaceKeyPrefixWith>  ]
      [ <ReplaceKeyWith>...</ReplaceKeyWith> ]
      [ <HttpRedirectCode>...</HttpRedirectCode> ]
   </Redirect>

Note: <Redirect> must have at least one child element. You can have either
 ReplaceKeyPrefix with or ReplaceKeyWith but not both.
```

Tabel berikut menjelaskan elemen-elemen dalam aturan perutean.

Nama	Deskripsi
`RoutingRules`	Kontainer untuk pengumpulan `RoutingRule` yang berbeda.
`RoutingRule`	Aturan yang mengidentifikasi kondisi dan pengalihan yang diterapkan saat kondisi terpenuhi. Ketentuan: • Kontainer `RoutingRules` harus memuat setidaknya satu aturan perutean.
`Condition`	Kontainer untuk menjelaskan kondisi yang harus dipenuhi untuk pengalihan yang ditentukan untuk diterapkan. Jika aturan perutean tidak mencakup syarat, aturan diterapkan ke semua permintaan.
`KeyPrefixEquals`	Prefiks nama kunci objek yang menjadi tujuan pengalihan permintaan. `KeyPrefixEquals` diperlukan jika `HttpErrorCodeReturnedEquals` tidak ditentukan. Jika `KeyPrefixEquals` dan `HttpErrorCodeReturnedEquals` ditentukan, keduanya harus bernilai benar agar syarat dapat dipenuhi.

Nama	Deskripsi
`HttpErrorCodeReturnedEquals`	Kode kesalahan HTTP yang harus cocok agar pengalihan dapat diterapkan. Jika terjadi kesalahan, dan jika kode kesalahan memenuhi nilai ini, maka pengalihan yang ditentukan berlaku. `HttpErrorCodeReturnedEquals` diperlukan jika `KeyPrefixEquals` tidak ditentukan. Jika `KeyPrefixEquals` dan `HttpErrorCodeReturnedEquals` ditentukan, keduanya harus bernilai benar agar syarat dapat dipenuhi.
`Redirect`	Elemen kontainer yang memberikan instruksi untuk mengalihkan permintaan. Anda dapat mengarahkan permintaan ke halaman host lain atau halaman lainnya, atau Anda dapat menentukan protokol lain yang akan digunakan. `RoutingRule` harus memiliki elemen `Redirect`. Elemen `Redirect` harus mengandung setidaknya satu dari elemen saudara berikut: `Protocol`, `HostName`, `ReplaceKeyPrefixWith`, `ReplaceKeyWith`, atau `HttpRedirectCode`.
`Protocol`	Protokolnya, `http` atau `https`, untuk digunakan dalam `Location` yang dikembalikan di respons. Jika salah satu dari saudaranya mendapat persediaan, `Protocol` tidak diperlukan.
`HostName`	Nama host yang akan digunakan dalam header `Location` yang dikembalikan dalam respons. Jika salah satu dari saudaranya mendapat persediaan, `HostName` tidak diperlukan.

Nama	Deskripsi
`ReplaceKeyPrefixWith`	Prefiks nama kunci objek yang menggantikan nilai `KeyPrefixEquals` dalam permintaan pengalihan. Jika salah satu dari saudaranya mendapat persediaan, `ReplaceKeyPrefixWith` tidak diperlukan. Perangkat ini hanya dapat disediakan jika `ReplaceKeyWith` tidak disediakan.
`ReplaceKeyWith`	Kunci objek yang akan digunakan dalam header `Location` yang dikembalikan dalam respons. Jika salah satu dari saudaranya mendapat persediaan, `ReplaceKeyWith` tidak diperlukan. Perangkat ini hanya dapat disediakan jika `ReplaceKeyPrefixWith` tidak disediakan.
`HttpRedirectCode`	Kode pengalihan HTTP yang akan digunakan dalam header `Location` yang dikembalikan dalam respons. Jika salah satu dari saudaranya mendapat persediaan, `HttpRedirectCode` tidak diperlukan.

Contoh aturan pengalihan

Contoh berikut ini menjelaskan tugas pengalihan umum:

> **Important**
>
> Untuk membuat aturan pengalihan di konsol Amazon S3, Anda harus menggunakan JSON.

Example 1: Alihkan setelah mengganti nama sebuah prefiks kunci

Misalkan bucket Anda berisi objek berikut:

- index.html
- docs/article1.html

- docs/article2.html

Anda memutuskan untuk mengganti nama folder dari `docs/` ke `documents/`. Setelah Anda membuat perubahan ini, Anda harus mengalihkan permintaan untuk prefiks `docs/` ke `documents/`. Misalnya, permintaan untuk `docs/article1.html` akan dialihkan ke `documents/article1.html`.

Dalam hal ini, Anda menambahkan aturan perutean berikut ke konfigurasi situs web.

JSON

```
[
    {
        "Condition": {
            "KeyPrefixEquals": "docs/"
        },
        "Redirect": {
            "ReplaceKeyPrefixWith": "documents/"
        }
    }
]
```

XML

```
  <RoutingRules>
    <RoutingRule>
    <Condition>
      <KeyPrefixEquals>docs/</KeyPrefixEquals>
    </Condition>
    <Redirect>
      <ReplaceKeyPrefixWith>documents/</ReplaceKeyPrefixWith>
    </Redirect>
    </RoutingRule>
  </RoutingRules>
```

Example 2: Alihkan permintaan untuk folder yang dihapus ke sebuah halaman

Misalkan Anda menghapus folder `images/` (yaitu, Anda menghapus semua objek dengan prefiks kunci `images/`). Anda dapat menambahkan aturan perutean yang mengalihkan permintaan untuk objek apa pun dengan prefiks kunci `images/` ke halaman yang diberi nama `folderdeleted.html`.

JSON

```
[
    {
        "Condition": {
            "KeyPrefixEquals": "images/"
        },
        "Redirect": {
            "ReplaceKeyWith": "folderdeleted.html"
        }
    }
]
```

XML

```
  <RoutingRules>
    <RoutingRule>
    <Condition>
       <KeyPrefixEquals>images/</KeyPrefixEquals>
    </Condition>
    <Redirect>
      <ReplaceKeyWith>folderdeleted.html</ReplaceKeyWith>
    </Redirect>
    </RoutingRule>
  </RoutingRules>
```

Example 3: Alihkan untuk kesalahan HTTP

Misalkan ketika objek yang diminta tidak ditemukan, Anda sebaiknya mengalihkan permintaan ke instans Amazon Elastic Compute Cloud (Amazon EC2). Tambahkan aturan pengalihan sehingga ketika kode status HTTP 404 (Tidak Ditemukan) dikembalikan, pengunjung situs dialihkan ke instans Amazon EC2 yang menangani permintaan tersebut.

Contoh berikut juga memasukkan prefiks kunci objek `report-404/` dalam pengalihan. Misalnya, jika Anda meminta halaman `ExamplePage.html` dan ini mengakibatkan kesalahan HTTP 404, permintaan akan diarahkan ke halaman `report-404/ExamplePage.html` pada instans Amazon EC2 yang ditentukan. Jika tidak ada aturan perutean dan terjadi kesalahan HTTP 404, dokumen kesalahan yang ditentukan dalam konfigurasi akan dikembalikan.

JSON

```
[
    {
        "Condition": {
            "HttpErrorCodeReturnedEquals": "404"
        },
        "Redirect": {
            "HostName": "ec2-11-22-333-44.compute-1.amazonaws.com",
            "ReplaceKeyPrefixWith": "report-404/"
        }
    }
]
```

XML

```
  <RoutingRules>
    <RoutingRule>
    <Condition>
      <HttpErrorCodeReturnedEquals>404</HttpErrorCodeReturnedEquals >
    </Condition>
    <Redirect>
      <HostName>ec2-11-22-333-44.compute-1.amazonaws.com</HostName>
      <ReplaceKeyPrefixWith>report-404/</ReplaceKeyPrefixWith>
    </Redirect>
    </RoutingRule>
  </RoutingRules>
```

Mengalihkan permintaan untuk objek

Anda dapat mengalihkan permintaan untuk objek ke objek atau URL lain dengan mengatur lokasi pengalihan situs web di metadata objek tersebut. Anda mengatur pengalihan dengan menambahkan properti `x-amz-website-redirect-location` ke metadata objek tersebut. Di konsol Amazon S3, Anda menetapkan Lokasi Pengalihan Situs Web dalam metadata objek tersebut. Jika Anda menggunakan API Amazon S3, Anda mengatur `x-amz-website-redirect-location`. Halaman web kemudian mengartikan objek sebagai pengalihan 301.

Untuk mengarahkan permintaan ke objek lain, Anda menetapkan lokasi pengalihan ke kunci objek target. Untuk mengalihkan permintaan ke URL eksternal, Anda mengatur lokasi pengalihan ke URL

yang diinginkan. Untuk informasi selengkapnya tentang pemberian metadata objek, lihat Metadata objek yang ditentukan sistem.

Saat Anda mengatur suatu halaman, Anda dapat menyimpan atau menghapus konten objek sumber. Misalnya, jika Anda memiliki objek `page1.html` di bucket Anda, Anda dapat mengalihkan permintaan apa pun untuk halaman ini ke objek lain, `page2.html`. Anda memiliki dua opsi:

- Simpan konten `page1.html` dan alihkan permintaan halaman.
- Hapus konten `page1.html` dan unggah objek zero-byte dengan nama `page1.html` untuk mengganti objek yang ada dan mengarahkan kembali permintaan halaman.

Menggunakan konsol S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket, pilih nama bucket yang telah Anda konfigurasikan sebagai situs web statis (misalnya, `example.com`).
3. Di Bawah Objek, pilih objek Anda.
4. Pilih Tindakan, lalu pilih Edit metadata.
5. Pilih Metadata.
6. Pilih Tambahkan Metadata.
7. Di Bawah Jenis, pilih Sistem Didefinisikan.
8. Di Key, pilih x-amz-website-redirect-location.
9. Di Nilai, masukkan nama kunci objek yang ingin Anda arahkan, misalnya, `/page2.html`.

   Untuk objek lain dalam bucket yang sama, prefiks `/` pada nilai wajib diisi. Anda juga dapat mengatur nilai ke URL eksternal, misalnya, `http://www.example.com`.
10. Pilih Edit metadata.

Penggunaan API REST

Tindakan API Amazon S3 berikut mendukung header `x-amz-website-redirect-location` dalam permintaan. Amazon S3 menyimpan nilai header di metadata objek sebagai `x-amz-website-redirect-location`.

- PUT Objek
- Mulai Unggahan Multibagian

- Objek POST
- PUT Objek-Salin

Bucket yang dikonfigurasi untuk hosting situs web memiliki titik akhir situs web dan titik akhir REST. Permintaan untuk halaman yang dikonfigurasi sebagai pengalihan 301 memiliki kemungkinan hasil berikut, tergantung pada titik akhir permintaan:

- Titik akhir situs web spesifik wilayah–Amazon S3 mengalihkan permintaan halaman sesuai dengan nilai properti `x-amz-website-redirect-location`.
- Titik akhir REST–Amazon S3 tidak mengalihkan permintaan halaman. Ini akan mengembalikan objek yang diminta.

Untuk informasi selengkapnya tentang titik akhir, lihat Perbedaan utama antara titik akhir situs web dan titik akhir API REST.

Saat mengatur pengalihan halaman, Anda dapat menyimpan atau menghapus konten objek. Misalnya, anggap bahwa Anda memiliki objek `page1.html` tersebut dalam bucket Anda.

- Untuk menyimpan konten `page1.html` dan hanya mengalihkan permintaan halaman, Anda dapat mengirimkan permintaan PUT Objek-Salin untuk membuat objek `page1.html` yang menggunakan objek `page1.html` yang sudah ada sebagai sumbernya. Dalam permintaan Anda, Anda menetapkan header `x-amz-website-redirect-location`. Ketika permintaan selesai, Anda akan memiliki halaman asli dengan konten yang tidak berubah, tetapi Amazon S3 akan mengalihkan semua permintaan halaman untuk halaman tersebut ke lokasi pengalihan yang Anda tentukan.
- Untuk menghapus konten objek `page1.html` dan mengalihkan permintaan untuk halaman, Anda dapat mengirim permintaan PUT Objek untuk mengunggah objek nol-byte yang memiliki kunci objek yang sama: `page1.html`. Dalam permintaan PUT, Anda menetapkan `x-amz-website-redirect-location` untuk `page1.html` ke objek baru. Setelah permintaan selesai, `page1.html` tidak memiliki konten, dan permintaan diarahkan kembali ke lokasi yang ditentukan oleh `x-amz-website-redirect-location`.

Saat Anda mengambil objek menggunakan tindakan GET Objek bersama dengan metadata objek lain, Amazon S3 mengembalikan header `x-amz-website-redirect-location` di dalam respons.

Mengembangkan dengan Amazon S3

Bagian ini mencakup topik terkait developer untuk menggunakan Amazon S3. Untuk informasi lebih lanjut, tinjau topik di bawah.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan direktori bucket, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Topik

- Membuat permintaan
- Mengembangkan dengan Amazon S3 menggunakan AWS CLI
- Mengembangkan dengan Amazon S3 menggunakan SDK AWS
- Berkembang dengan Amazon S3 menggunakan API REST
- Menangani kesalahan REST dan SOAP
- Referensi developer

Membuat permintaan

Amazon S3 adalah layanan REST. Anda dapat mengirim permintaan ke Amazon S3 menggunakan REST API atau AWS SDK (lihat Kode Sampel dan Pustaka) yang membungkus REST API Amazon S3 mendasar, yang menyederhanakan tugas pemrograman Anda.

Setiap interaksi dengan Amazon S3 diautentikasi atau dilakukan secara anonim. Autentikasi adalah proses verifikasi identitas pemohon yang mencoba mengakses produk Amazon Web Services (AWS). Permintaan yang diautentikasi harus menyertakan nilai tanda tangan yang mengautentikasi pengirim permintaan. Nilai tanda tangan, sebagian, dihasilkan dari kunci akses AWS pemohon (ID kunci akses dan kunci akses rahasia). Untuk informasi lebih lanjut tentang mendapatkan kunci akses, lihat Bagaimana Saya Mendapatkan Kredensi Keamanan? di Referensi Umum AWS.

Jika Anda menggunakan AWS SDK, pustaka menghitung tanda tangan dari kunci yang Anda sediakan. Namun, jika Anda melakukan panggilan langsung API REST di aplikasi Anda, Anda harus menulis kode untuk menghitung tanda tangan dan menambahkannya ke permintaan.

Topik

- Tentang kunci akses
- Meminta titik akhir
- Membuat permintaan ke Amazon S3 melalui IPv6
- Membuat permintaan menggunakan AWS SDK
- Membuat permintaan menggunakan API REST

Tentang kunci akses

Bagian berikut meninjau jenis kunci akses yang dapat Anda gunakan untuk membuat permintaan terautentikasi.

Tombol akses Akun AWS

Kunci akses akun memberikan akses penuh ke sumber daya AWS yang dimiliki oleh akun tersebut. Berikut ini adalah contoh kunci akses:

- ID kunci akses (20 karakter, string alfanumerik). Sebagai contoh: AKIAIOSFODNN7EXAMPLE
- Kunci akses rahasia (string 40 karakter). Misalnya:bPxRfi wJalrXUtnFEMI/K7MDENG/cyPLEKEY

ID kunci akses secara unik mengidentifikasi Akun AWS. Anda dapat menggunakan kunci akses ini untuk mengirim permintaan terautentikasi ke Amazon S3.

Kunci akses pengguna IAM

Anda dapat membuat satu Akun AWS untuk perusahaan Anda; tetapi, mungkin terdapat beberapa karyawan dalam organisasi yang memerlukan akses ke sumber daya AWS organisasi Anda. Berbagi kunci akses Akun AWS Anda mengurangi keamanan, dan membuat Akun AWS individu untuk setiap karyawan mungkin tidak praktis. Selain itu, Anda tidak dapat membagikan sumber daya seperti bucket dan objek dengan mudah karena itu dimiliki oleh akun yang berbeda. Untuk berbagi sumber daya, Anda harus memberikan izin, yang merupakan pekerjaan tambahan.

Dalam skenario tersebut, Anda dapat menggunakanAWS Identity and Access Management (IAM) untuk membuat pengguna dalam AndaAkun AWS dengan kunci akses mereka sendiri dan melampirkan kebijakan pengguna IAM yang memberikan izin akses sumber daya yang sesuai kepada pengguna tersebut. Untuk mengelola pengguna ini dengan lebih baik, IAM memungkinkan

Anda membuat kelompok pengguna dan memberikan izin tingkat kelompok yang berlaku bagi semua pengguna dalam grup tersebut.

Pengguna ini disebut sebagai pengguna IAM yang Anda buat dan kelola di dalam AWS. Akun induk mengontrol kemampuan pengguna untuk mengakses AWS. Setiap sumber daya yang dibuat pengguna IAM dikendalikan dan dibayar oleh akun Akun AWS. Pengguna IAM ini dapat mengirimkan permintaan terautentikasi ke Amazon S3 menggunakan kredensial keamanan mereka sendiri. Untuk informasi lebih lanjut tentang membuat dan mengelola pengguna di bawah Akun AWS Anda, kunjungi halaman detail produk AWS Identity and Access Management.

Kredensial keamanan sementara

Selain membuat pengguna IAM dengan kunci akses mereka sendiri, IAM juga memungkinkan Anda memberikan kredensial keamanan sementara (kunci akses sementara dan token keamanan) kepada setiap pengguna IAM agar mereka dapat mengakses layanan dan sumber daya AWS Anda. Anda juga dapat mengelola pengguna dalam sistem Anda di luar AWS. Ini disebut sebagai pengguna gabungan. Selain itu, pengguna dapat menjadi aplikasi yang Anda buat untuk mengakses sumber daya AWS Anda.

IAM menyediakan API AWS Security Token Service agar Anda dapat meminta kredensial keamanan sementara. Anda dapat menggunakan API AWS STS atau AWS SDK untuk meminta kredensial ini. API mengembalikan kredensial keamanan sementara (ID kunci kunci akses dan kunci akses rahasia), dan token keamanan. Kredensial ini hanya valid untuk durasi yang Anda tentukan saat Anda memintanya. Anda menggunakan ID kunci akses dan kunci rahasia dengan cara yang sama ketika mengirim permintaan menggunakan Akun AWS Anda atau kunci akses pengguna IAM Anda. Selain itu, Anda harus menyertakan token dalam setiap permintaan yang Anda kirim ke Amazon S3.

Pengguna IAM dapat meminta kredensial keamanan sementara ini untuk penggunaan mereka sendiri atau memberikannya kepada pengguna atau aplikasi gabungan. Saat meminta kredensial keamanan sementara untuk pengguna gabungan, Anda harus memberikan nama pengguna dan kebijakan IAM yang menentukan izin yang ingin Anda kaitkan dengan kredensial keamanan sementara ini. Pengguna gabungan tidak dapat memperoleh izin lebih dari pengguna IAM induk yang meminta kredensial sementara.

Anda dapat menggunakan kredensial keamanan sementara ini dalam mengajukan permintaan kepada Amazon S3. Pustaka API menghitung nilai tanda tangan yang diperlukan menggunakan kredensial tersebut untuk mengautentikasi permintaan Anda. Jika Anda mengirim permintaan menggunakan kredensial kedaluwarsa, Amazon S3 menolak permintaan tersebut.

Untuk informasi tentang penandatanganan permintaan menggunakan kredensial keamanan sementara di permintaan API REST Anda, lihat Menandatangani dan mengautentikasi permintaan REST. Untuk informasi tentang mengirim permintaan menggunakan AWS SDK, lihat Membuat permintaan menggunakan AWS SDK.

Untuk informasi lebih lanjut tentang dukungan IAM untuk kredensial keamanan sementara, lihat Kredensial Keamanan Sementara dalam Panduan Pengguna IAM.

Untuk keamanan tambahan, Anda dapat mengharuskan autentikasi multifaktor (MFA) ketika mengakses sumber daya Amazon S3 Anda dengan mengonfigurasi kebijakan bucket. Untuk informasi, lihat Membutuhkan MFA. Setelah Anda mengharuskan MFA untuk mengakses sumber daya Amazon S3, satu-satunya cara Anda untuk mengakses sumber daya ini adalah dengan memberikan kredensial sementara yang dibuat dengan kunci MFA. Untuk informasi lebih lanjut, lihat halaman detail Multi-Factor Authentication AWS dan Mengonfigurasi Akses API yang Dilindungi MFA dalam Panduan Pengguna IAM.

Meminta titik akhir

Anda mengirimkan permintaan REST ke titik akhir layanan yang telah ditentukan sebelumnya. Untuk daftar semuaAWS layanan dan titik akhir terkait, kunjungi Wilayah dan Titik Akhir dalam Referensi Umum AWS.

Membuat permintaan ke Amazon S3 melalui IPv6

Amazon Simple Storage Service (Amazon S3) mendukung kemampuan untuk mengakses bucket S3 menggunakan Protokol Internet versi 6 (IPv6), selain protokol IPv4. Titik akhir tumpukan ganda Amazon S3 mendukung permintaan ke bucket S3 melalui IPv6 dan IPv4. Tidak ada biaya tambahan untuk mengakses Amazon S3 melalui IPv6. Untuk informasi selengkapnya tentang harga, lihat Harga Amazon S3.

Topik

- Memulai membuat permintaan melalui IPv6
- Menggunakan alamat IPv6 di kebijakan IAM
- Menguji kompatibilitas alamat IP
- Menggunakan titik akhir tumpukan ganda Amazon S3

Memulai membuat permintaan melalui IPv6

Untuk mengajukan permintaan ke bucket S3 melalui IPv6, Anda perlu menggunakan titik akhir tumpukan ganda. Bagian selanjutnya menjelaskan cara untuk membuat permintaan melalui IPv6 dengan menggunakan titik akhir tumpukan ganda.

Berikut adalah beberapa hal yang harus Anda ketahui sebelum mencoba mengakses bucket melalui IPv6:

- Klien dan jaringan yang mengakses bucket harus diaktifkan agar dapat menggunakan IPv6.
- Permintaan cara hosting virtual dan cara jejak didukung untuk akses IPv6. Untuk informasi selengkapnya, lihat Titik akhir tumpukan ganda Amazon S3.
- Jika Anda menggunakan pemfilteran alamat IP sumber dalam kebijakan pengguna atau bucket AWS Identity and Access Management (IAM), Anda perlu memperbarui kebijakan untuk menyertakan rentang alamat IPv6. Untuk informasi selengkapnya, lihat Menggunakan alamat IPv6 di kebijakan IAM.
- Saat menggunakan IPv6, berkas log akses server mengeluarkan alamat IP dalam format IPv6. Anda perlu memperbarui alat, skrip, dan perangkat lunak yang ada yang Anda gunakan untuk mengurai file log Amazon S3 sehingga mereka dapat mengurai alamat `Remote IP` yang diformat IPv6. Untuk informasi lebih lanjut, lihat Server Amazon S3 mengakses format log dan Pencatatan permintaan dengan pencatatan akses server.

> **Note**
>
> Jika Anda mengalami masalah terkait dengan kehadiran alamat IPv6 dalam berkas log, hubungi AWS Support.

Membuat permintaan melalui IPv6 dengan menggunakan titik akhir tumpukan ganda

Anda membuat permintaan dengan panggilan API Amazon S3 melalui IPv6 dengan menggunakan titik akhir tumpukan ganda. Operasi API Amazon S3 bekerja dengan cara yang sama baik saat Anda mengakses Amazon S3 melalui IPv6 atau melalui IPv4. Begitupun dengan kinerja.

Saat menggunakan API REST, Anda mengakses langsung titik akhir tumpukan ganda. Untuk informasi selengkapnya, lihat Titik akhir tumpukan ganda.

Saat menggunakan AWS Command Line Interface (AWS CLI) dan AWS SDK, Anda dapat menggunakan parameter atau flag untuk mengubah ke titik akhir dual-stack. Anda juga dapat menentukan titik akhir tumpukan ganda secara langsung sebagai pembatalan titik akhir Amazon S3 dalam berkas config.

Anda dapat menggunakan titik akhir tumpukan ganda untuk mengakses bucket melalui IPv6 dari yang berikut ini:

- Itu AWS CLI, lihatMenggunakan titik akhir tumpukan ganda dari AWS CLI.
- AWS SDK, lihatMenggunakan titik akhir tumpukan ganda dari AWS SDK.
- API REST, lihat Membuat permintaan ke titik akhir dual-stack dengan menggunakan API REST.

Fitur tidak tersedia melalui IPv6

Fitur berikut saat ini tidak didukung ketika mengakses bucket S3 melalui IPv6: Hosting situs web statis dari bucket S3.

Menggunakan alamat IPv6 di kebijakan IAM

Sebelum mencoba mengakses bucket menggunakan IPv6, Anda harus memastikan bahwa setiap pengguna IAM atau kebijakan bucket S3 yang digunakan untuk pemfilteran alamat IP diperbarui untuk menyertakan rentang alamat IPv6. Kebijakan pemfilteran alamat IP yang tidak diperbarui untuk ditangani alamat IPv6 dapat menyebabkan klien kehilangan atau mendapatkan akses ke bucket dengan tidak benar saat mereka mulai menggunakan IPv6. Untuk informasi selengkapnya tentang mengelola izin akses, lihat Identity and Access Management untuk Amazon S3.

Kebijakan IAM yang memfilter alamat IP menggunakan Operator Kondisi Alamat IP. Kebijakan bucket berikut mengidentifikasi rentang 54.240.143.* yang diizinkan alamat IPv4 dengan menggunakan operator kondisi alamat IP. Setiap alamat IP di luar rentang ini akan ditolak akses ke bucket (`examplebucket`). Karena semua alamat IPv6 di luar rentang yang diizinkan, kebijakan ini mencegah alamat IPv6 agar tidak dapat mengakses `examplebucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
```

```
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::examplebucket/*",
      "Condition": {
         "IpAddress": {"aws:SourceIp": "54.240.143.0/24"}
      }
    }
  ]
}
```

Anda dapat memodifikasi elemen `Condition` kebijakan bucket yang memungkinkan rentang alamat IPv4 (`54.240.143.0/24`) dan IPv6 (`2001:DB8:1234:5678::/64`) seperti yang ditunjukkan dalam contoh berikut. Anda dapat menggunakan tipe blok `Condition` yang ditampilkan dalam contoh untuk memperbarui kebijakan pengguna IAM dan bucket Anda.

```
       "Condition": {
         "IpAddress": {
            "aws:SourceIp": [
              "54.240.143.0/24",
               "2001:DB8:1234:5678::/64"
             ]
          }
        }
```

Sebelum menggunakan IPv6 Anda harus memperbarui semua kebijakan pengguna IAM dan bucket terkait yang menggunakan pemfilteran alamat IP untuk memungkinkan rentang alamat IPv6 . Kami menyarankan agar Anda memperbarui kebijakan IAM dengan rentang alamat IPv6 organisasi Anda selain dari rentang alamat IPv4 Anda yang telah ada. Untuk contoh kebijakan bucket yang memungkinkan akses atas IPv6 dan IPv4, lihat Membatasi akses ke suatu Wilayah tertentu.

Anda dapat meninjau kebijakan pengguna IAM menggunakan konsol IAM di https://console.aws.amazon.com/iam/. Untuk informasi lebih lanjut tentang IAM, lihat Panduan Pengguna IAM. Untuk informasi lebih lanjut tentang kebijakan bucket S3, lihat Menambahkan kebijakan bucket dengan menggunakan konsol Amazon S3.

Menguji kompatibilitas alamat IP

Jika Anda menggunakan Linux/Unix atau Mac OS X, Anda dapat menguji apakah Anda dapat mengakses titik akhir tumpukan ganda melalui IPv6 dengan menggunakan perintah `curl` seperti yang ditunjukkan dalam contoh berikut:

Example

```
curl -v  http://s3.dualstack.us-west-2.amazonaws.com/
```

Anda mendapatkan informasi yang serupa dengan contoh berikut. Jika Anda terhubung melalui IPv6, alamat IP yang terhubung akan menjadi alamat IPv6.

```
* About to connect() to s3-us-west-2.amazonaws.com port 80 (#0)
*   Trying IPv6 address... connected
* Connected to s3.dualstack.us-west-2.amazonaws.com (IPv6 address) port 80 (#0)
> GET / HTTP/1.1
> User-Agent: curl/7.18.1 (x86_64-unknown-linux-gnu) libcurl/7.18.1 OpenSSL/1.0.1t
 zlib/1.2.3
> Host: s3.dualstack.us-west-2.amazonaws.com
```

Jika Anda menggunakan Microsoft Windows 7 atau Windows 10, Anda dapat menguji apakah Anda dapat mengakses titik akhir tumpukan ganda melalui IPv6 atau IPv4 dengan menggunakan perintah `ping` seperti yang diperlihatkan dalam contoh berikut.

```
ping ipv6.s3.dualstack.us-west-2.amazonaws.com
```

Menggunakan titik akhir tumpukan ganda Amazon S3

Titik akhir tumpukan ganda Amazon S3 mendukung permintaan ke bucket S3 melalui IPv6 dan IPv4. Bagian ini menjelaskan cara untuk menggunakan titik akhir tumpukan ganda.

Topik

- Titik akhir tumpukan ganda Amazon S3
- Menggunakan titik akhir tumpukan ganda dari AWS CLI
- Menggunakan titik akhir tumpukan ganda dari AWS SDK
- Menggunakan titik akhir tumpukan ganda dari API REST

Titik akhir tumpukan ganda Amazon S3

Saat Anda membuat permintaan ke titik akhir tumpukan ganda, URL bucket memutuskan ke alamat IPv6 atau IPv4. Untuk informasi lebih lanjut tentang mengakses bucket melalui IPv6, lihat Membuat permintaan ke Amazon S3 melalui IPv6.

Saat menggunakan API REST, Anda secara langsung mengakses titik akhir Amazon S3 dengan menggunakan nama titik akhir (URI). Anda dapat mengakses bucket S3 melalui titik akhir tumpukan ganda dengan menggunakan nama titik akhir cara hosting virtual dan cara jejak. Amazon S3 hanya mendukung nama titik akhir tumpukan ganda regional, yang berarti Anda harus menentukan wilayah sebagai bagian dari nama.

Gunakan konvensi penamaan berikut untuk nama titik akhir cara hosting virtual dan cara jejak:

- Titik akhir tumpukan ganda cara hosting virtual:

  bucketname.s3.dualstack.*aws-region*.amazonaws.com

- Titik akhir tumpukan ganda cara jejak:

  s3.dualstack.*aws-region*.amazonaws.com/*bucketname*

Untuk informasi lebih lanjut, tentang cara penamaan titik akhir, lihat Mengakses dan mendaftarkan bucket Amazon S3. Untuk daftar titik akhir Amazon S3, lihat Wilayah dan Titik Akhir di Referensi Umum AWS.

> **Important**
>
> Anda dapat menggunakan akselerasi transfer dengan titik akhir tumpukan ganda. Untuk informasi selengkapnya, lihat Memulai Amazon S3 Transfer Acceleration.

> **Note**
>
> Dua jenis titik akhir VPC untuk mengakses Amazon S3 (titik akhir VPC Antarmuka dan titik akhir VPC Gateway) tidak memiliki dukungan dual-stack. Untuk informasi selengkapnya tentang titik akhir VPC untuk Amazon S3, lihat. AWS PrivateLink untuk Amazon S3

Saat menggunakan AWS Command Line Interface (AWS CLI) dan AWS SDK, Anda dapat menggunakan parameter atau flag untuk mengubah ke titik akhir dual-stack. Anda juga dapat menentukan titik akhir tumpukan ganda secara langsung sebagai pembatalan titik akhir Amazon S3

dalam berkas config. Bagian berikut menjelaskan cara menggunakan titik akhir dual-stack dari AWS CLI dan SDK. AWS

Menggunakan titik akhir tumpukan ganda dari AWS CLI

Bagian ini memberikan contoh AWS CLI perintah yang digunakan untuk membuat permintaan ke endpoint dual-stack. Untuk petunjuk tentang pengaturan AWS CLI, lihatMengembangkan dengan Amazon S3 menggunakan AWS CLI.

Anda menetapkan nilai konfigurasi `use_dualstack_endpoint` ke `true` dalam profil dalam AWS Config file Anda untuk mengarahkan semua permintaan Amazon S3 yang dibuat oleh `s3api` AWS CLI perintah `s3` dan ke titik akhir tumpukan ganda untuk wilayah yang ditentukan. Anda menentukan wilayah dalam berkas config atau dalam perintah menggunakan opsi `--region`.

Saat menggunakan titik akhir dual-stack dengan AWS CLI, keduanya `path` dan gaya `virtual` pengalamatan didukung. Cara pengalamatan, diatur dalam berkas config, mengontrol apakah nama bucket ada dalam nama host atau bagian dari URL. Secara default, CLI akan mencoba menggunakan cara virtual jika memungkinkan, tetapi akan kembali ke cara jejak jika perlu. Untuk informasi selengkapnya, lihat Konfigurasi Amazon S3 AWS CLI.

Anda juga dapat membuat perubahan konfigurasi dengan menggunakan perintah, seperti ditunjukkan dalam contoh berikut, yang menetapkan `use_dualstack_endpoint` ke `true` dan `addressing_style` ke `virtual` di profil default.

```
$ aws configure set default.s3.use_dualstack_endpoint true
$ aws configure set default.s3.addressing_style virtual
```

Jika Anda ingin menggunakan titik akhir dual-stack hanya untuk AWS CLI perintah tertentu (tidak semua perintah), Anda dapat menggunakan salah satu dari metode berikut:

- Anda dapat menggunakan titik akhir tumpukan ganda sesuai perintah dengan mengatur parameter `--endpoint-url` ke `https://s3.dualstack.`*`aws-region`*`.amazonaws.com` atau `http://s3.dualstack.`*`aws-region`*`.amazonaws.com` untuk setiap `s3` atau perintah `s3api`.

  ```
  $ aws s3api list-objects --bucket bucketname --endpoint-url https://s3.dualstack.aws-region.amazonaws.com
  ```

- Anda dapat mengatur profil terpisah di AWS Config file Anda. Sebagai contoh, buat satu profil yang menetapkan `use_dualstack_endpoint` ke `true` dan profil yang tidak mengatur `use_dualstack_endpoint`. Ketika Anda menjalankan perintah, tentukan profil mana yang ingin

Anda gunakan, tergantung pada apakah Anda ingin menggunakan titik akhir tumpukan ganda atau tidak.

> **Note**
>
> Saat menggunakan, AWS CLI Anda saat ini tidak dapat menggunakan akselerasi transfer dengan titik akhir dual-stack. Namun, dukungan untuk akan segera hadir. AWS CLI Untuk informasi selengkapnya, lihat Menggunakan AWS CLI.

Menggunakan titik akhir tumpukan ganda dari AWS SDK

Bagian ini memberikan contoh cara mengakses titik akhir dual-stack dengan menggunakan SDK. AWS

AWS SDK for Java contoh titik akhir tumpukan ganda

Contoh berikut menunjukkan cara mengaktifkan titik akhir tumpukan ganda saat membuat klien Amazon S3 menggunakan AWS SDK for Java.

Untuk petunjuk cara membuat dan menguji sampel Java yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;

public class DualStackEndpoints {

    public static void main(String[] args) {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            // Create an Amazon S3 client with dual-stack endpoints enabled.
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
```

```
                    .withRegion(clientRegion)
                    .withDualstackEnabled(true)
                    .build();

            s3Client.listObjects(bucketName);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Jika Anda menggunakan AWS SDK for Java pada Windows, Anda mungkin harus mengatur properti Java virtual machine (JVM) berikut:

```
java.net.preferIPv6Addresses=true
```

AWS Contoh titik akhir tumpukan ganda .NET SDK

Saat menggunakan AWS SDK untuk.NET Anda menggunakan `AmazonS3Config` kelas untuk mengaktifkan penggunaan titik akhir dual-stack seperti yang ditunjukkan pada contoh berikut.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class DualStackEndpointTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
        private static IAmazonS3 client;
```

```
        public static void Main()
        {
            var config = new AmazonS3Config
            {
                UseDualstackEndpoint = true,
                RegionEndpoint = bucketRegion
            };
            client = new AmazonS3Client(config);
            Console.WriteLine("Listing objects stored in a bucket");
            ListingObjectsAsync().Wait();
        }

        private static async Task ListingObjectsAsync()
        {
            try
            {
                var request = new ListObjectsV2Request
                {
                    BucketName = bucketName,
                    MaxKeys = 10
                };
                ListObjectsV2Response response;
                do
                {
                    response = await client.ListObjectsV2Async(request);

                    // Process the response.
                    foreach (S3Object entry in response.S3Objects)
                    {
                        Console.WriteLine("key = {0} size = {1}",
                            entry.Key, entry.Size);
                    }
                    Console.WriteLine("Next Continuation Token: {0}",
response.NextContinuationToken);
                    request.ContinuationToken = response.NextContinuationToken;
                } while (response.IsTruncated == true);
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine("An AmazonS3Exception was thrown. Exception: " +
amazonS3Exception.ToString());
            }
            catch (Exception e)
            {
```

```
                Console.WriteLine("Exception: " + e.ToString());
            }
        }
    }
}
```

Untuk sampel .NET penuh untuk objek daftar, lihat Membuat daftar kunci objek secara terprogram.

Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

Menggunakan titik akhir tumpukan ganda dari API REST

Untuk informasi tentang permintaan sampai ke titik akhir tumpukan ganda dengan menggunakan API REST, lihat Membuat permintaan ke titik akhir dual-stack dengan menggunakan API REST.

Membuat permintaan menggunakan AWS SDK

Topik

- Membuat permintaan menggunakan Akun AWS atau kredensil pengguna IAM
- Membuat permintaan menggunakan kredensial sementara pengguna IAM
- Membuat permintaan menggunakan kredensial sementara pengguna gabungan

Anda dapat mengirim permintaan terautentikasi ke Amazon S3 menggunakan AWS SDK atau dengan melakukan panggilan REST API secara langsung di aplikasi Anda. API AWS SDK menggunakan kredensial yang Anda berikan untuk menghitung tanda tangan untuk autentikasi. Jika Anda menggunakan API REST secara langsung di aplikasi Anda, Anda harus menulis kode yang diperlukan untuk menghitung tanda tangan untuk mengautentikasi permintaan Anda. Untuk daftar AWS SDK yang tersedia buka, Kode Sampel dan Pustaka.

Membuat permintaan menggunakan Akun AWS atau kredensil pengguna IAM

Anda dapat menggunakan kredensil keamanan pengguna IAM Akun AWS atau IAM untuk mengirim permintaan yang diautentikasi ke Amazon S3. Bagian ini memberikan contoh bagaimana Anda dapat mengirim permintaan yang diautentikasi menggunakan AWS SDK for Java, AWS SDK for .NET, dan AWS SDK for PHP. Untuk daftar AWS SDK yang tersedia, buka Kode Sampel dan Pustaka.

Masing-masing AWS SDK ini menggunakan rantai penyedia kredensial khusus SDK untuk menemukan dan menggunakan kredensil serta melakukan tindakan atas nama pemilik kredensil. Kesamaan dari semua rantai penyedia kredensi ini adalah bahwa mereka semua mencari file AWS kredensional lokal Anda.

Untuk informasi selengkapnya, lihat topik di bawah:

Topik

- Untuk membuat file AWS kredensial lokal
- Mengirim permintaan yang diautentikasi menggunakan SDK AWS
- Sumber daya terkait

Untuk membuat file AWS kredensial lokal

Cara termudah untuk mengonfigurasi kredensil untuk AWS SDK Anda adalah dengan menggunakan file kredensial. AWS Jika Anda menggunakan AWS Command Line Interface (AWS CLI), Anda

mungkin sudah memiliki file AWS kredensial lokal yang dikonfigurasi. Jika tidak, gunakan prosedur berikut untuk mengatur file kredensial:

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Buat pengguna baru dengan izin yang terbatas pada layanan dan tindakan yang Anda inginkan untuk dapat diakses oleh kode Anda. Untuk informasi selengkapnya tentang membuat pengguna baru, lihat Membuat pengguna IAM (Konsol), dan ikuti instruksi hingga langkah 8.
3. Pilih Unduh .csv untuk menyimpan salinan lokal kredensial AWS Anda.
4. Pada komputer Anda, navigasi ke direktori beranda Anda, dan buat direktori `.aws`. Pada sistem berbasis Unix, seperti Linux atau OS X, ini berada di lokasi berikut:

   ```
   ~/.aws
   ```

   Di Windows, ini ada di lokasi berikut:

   ```
   %HOMEPATH%\.aws
   ```

5. Di direktori `.aws`, buat file baru bernama `credentials`.
6. Buka file `.csv` kredensial yang Anda unduh dari konsol IAM, dan salin isinya ke file `credentials` menggunakan format berikut:

   ```
   [default]
   aws_access_key_id = your_access_key_id
   aws_secret_access_key = your_secret_access_key
   ```

7. Simpan file `credentials`, dan hapus file `.csv` yang Anda unduh di langkah 3.

File kredensial bersama Anda sekarang dikonfigurasi di komputer lokal Anda, dan siap digunakan dengan SDK AWS .

Mengirim permintaan yang diautentikasi menggunakan SDK AWS

Gunakan AWS SDK untuk mengirim permintaan yang diautentikasi. Untuk informasi selengkapnya tentang mengirim permintaan terautentikasi, lihat AWS Kredensial keamanan atau Autentikasi Pusat Identitas IAM.

Java

Untuk mengirim permintaan yang diautentikasi ke Amazon S3 menggunakan kredensil pengguna atau IAM Akun AWS Anda, lakukan hal berikut:

- Gunakan kelas `AmazonS3ClientBuilder` untuk membuat instans `AmazonS3Client`.
- Jalankan salah satu metode `AmazonS3Client` untuk mengirim permintaan ke Amazon S3. Klien membuat tanda tangan yang diperlukan dari kredensial yang Anda berikan dan menyertakannya dalam permintaan.

Contoh berikut melakukan tugas sebelumnya. Untuk informasi tentang membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

Example

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListObjectsRequest;
import com.amazonaws.services.s3.model.ObjectListing;
import com.amazonaws.services.s3.model.S3ObjectSummary;

import java.io.IOException;
import java.util.List;

public class MakingRequests {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Bucket name ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Get a list of objects in the bucket, two at a time, and
```

```
            // print the name and size of each object.
            ListObjectsRequest listRequest = new
 ListObjectsRequest().withBucketName(bucketName).withMaxKeys(2);
            ObjectListing objects = s3Client.listObjects(listRequest);
            while (true) {
                List<S3ObjectSummary> summaries = objects.getObjectSummaries();
                for (S3ObjectSummary summary : summaries) {
                    System.out.printf("Object \"%s\" retrieved with size %d\n",
 summary.getKey(), summary.getSize());
                }
                if (objects.isTruncated()) {
                    objects = s3Client.listNextBatchOfObjects(objects);
                } else {
                    break;
                }
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Untuk mengirim permintaan yang diautentikasi menggunakan kredensi pengguna Akun AWS atau IAM Anda:

- Buat instans dari kelas `AmazonS3Client`.
- Jalankan salah satu metode `AmazonS3Client` untuk mengirim permintaan ke Amazon S3. Klien menghasilkan tanda tangan yang diperlukan dari kredensial yang Anda berikan dan menyertakannya dalam permintaan yang dikirim ke Amazon S3.

Untuk informasi selengkapnya, lihat Membuat permintaan menggunakan Akun AWS atau kredensil pengguna IAM.

> **Note**
>
> - Anda dapat membuat klien `AmazonS3Client` tanpa memberikan kredensial keamanan Anda. Permintaan yang dikirim menggunakan klien ini adalah permintaan anonim, tanpa tanda tangan. Amazon S3 menampilkan pesan kesalahan jika Anda mengirim permintaan anonim untuk sumber daya yang tidak tersedia untuk umum.
> - Anda dapat membuat Akun AWS dan membuat pengguna yang diperlukan. Anda juga dapat mengelola kredensial untuk pengguna tersebut. Anda perlu kredensial ini untuk melakukan tugas dalam contoh berikut. Untuk informasi selengkapnya, lihat Mengonfigurasi kredensial AWS di Panduan Developer AWS SDK for .NET .
>
> Anda kemudian juga dapat mengonfigurasi aplikasi Anda untuk secara aktif mengambil profil dan kredensil, dan kemudian secara eksplisit menggunakan kredensil tersebut saat membuat klien layanan. AWS Untuk informasi selengkapnya, lihat Mengakses kredensial dan profil dalam aplikasi di Panduan Developer AWS SDK for .NET .

Contoh C# berikut menunjukkan cara melakukan tugas sebelumnya. Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

Example

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class MakeS3RequestTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
```

```
        {
            using (client = new AmazonS3Client(bucketRegion))
            {
                Console.WriteLine("Listing objects stored in a bucket");
                ListingObjectsAsync().Wait();
            }
        }

        static async Task ListingObjectsAsync()
        {
            try
            {
                ListObjectsRequest request = new ListObjectsRequest
                {
                    BucketName = bucketName,
                    MaxKeys = 2
                };
                do
                {
                    ListObjectsResponse response = await
 client.ListObjectsAsync(request);
                    // Process the response.
                    foreach (S3Object entry in response.S3Objects)
                    {
                        Console.WriteLine("key = {0} size = {1}",
                            entry.Key, entry.Size);
                    }

                    // If the response is truncated, set the marker to get the next
                    // set of keys.
                    if (response.IsTruncated)
                    {
                        request.Marker = response.NextMarker;
                    }
                    else
                    {
                        request = null;
                    }
                } while (request != null);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
```

```
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
        }
    }
}
```

Untuk contoh pekerjaan, lihat Gambaran umum objek Amazon S3 dan Gambaran umum bucket. Anda dapat menguji contoh-contoh ini menggunakan kredensi pengguna IAM Akun AWS atau IAM Anda.

Misalnya, untuk mencantumkan semua kunci objek di bucket Anda, lihat Membuat daftar kunci objek secara terprogram.

PHP

Bagian ini menjelaskan cara menggunakan kelas dari versi 3 AWS SDK for PHP untuk mengirim permintaan yang diautentikasi menggunakan kredensil pengguna Anda Akun AWS atau IAM. Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

Contoh PHP berikut menunjukkan bagaimana klien mengajukan permintaan menggunakan kredensial keamanan Anda untuk mencantumkan semua bucket untuk akun Anda.

Example

```
require 'vendor/autoload.php';

use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;

$bucket = '*** Your Bucket Name ***';

$s3 = new S3Client([
    'region' => 'us-east-1',
    'version' => 'latest',
]);

// Retrieve the list of buckets.
```

```
$result = $s3->listBuckets();

try {
    // Retrieve a paginator for listing objects.
    $objects = $s3->getPaginator('ListObjects', [
        'Bucket' => $bucket
    ]);

    echo "Keys retrieved!" . PHP_EOL;

    // Print the list of objects to the page.
    foreach ($objects as $object) {
        echo $object['Key'] . PHP_EOL;
    }
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

> **Note**
>
> Anda dapat membuat klien `S3Client` tanpa memberikan kredensial keamanan Anda. Permintaan yang dikirim menggunakan klien ini adalah permintaan anonim, tanpa tanda tangan. Amazon S3 menampilkan pesan kesalahan jika Anda mengirim permintaan anonim untuk sumber daya yang tidak tersedia untuk umum. Untuk informasi selengkapnya, lihat Membuat Klien Anonim dalam Dokumentasi AWS SDK for PHP.

Untuk contoh pekerjaan, lihat Gambaran umum objek Amazon S3. Anda dapat menguji contoh-contoh ini menggunakan kredensil pengguna Akun AWS atau IAM Anda.

Untuk contoh kunci objek daftar dalam bucket, lihat Membuat daftar kunci objek secara terprogram.

Ruby

Sebelum Anda dapat menggunakan versi 3 AWS SDK for Ruby untuk melakukan panggilan ke Amazon S3, Anda harus menyetel kredenal AWS akses yang digunakan SDK untuk memverifikasi akses ke bucket dan objek Anda. Jika Anda telah berbagi kredensil yang diatur dalam profil AWS kredensial di sistem lokal Anda, versi 3 SDK for Ruby dapat menggunakan kredensialnya tanpa harus mendeklarasikannya dalam kode Anda. Untuk informasi selengkapnya tentang

pengaturan kredensial bersama, lihat Membuat permintaan menggunakan Akun AWS atau kredensil pengguna IAM.

Cuplikan kode Ruby berikut menggunakan kredensil dalam file AWS kredensi bersama di komputer lokal untuk mengautentikasi permintaan agar mendapatkan semua nama kunci objek dalam bucket tertentu. Contoh ini melakukan hal-hal berikut:

1. Buat instans dari kelas `Aws::S3::Client`.
2. Membuat permintaan ke Amazon S3 dengan menghitung objek dalam bucket menggunakan metode `list_objects_v2` dari `Aws::S3::Client`. Klien menghasilkan nilai tanda tangan yang diperlukan dari kredensil dalam file AWS kredensil di komputer Anda, dan memasukkannya ke dalam permintaan yang dikirimkannya ke Amazon S3.
3. Mencetak susunan nama kunci objek ke terminal.

Example

```
# Prerequisites:
#  - An existing Amazon S3 bucket.

require "aws-sdk-s3"

# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The bucket's name.
# @return [Boolean] true if all operations succeed; otherwise, false.
# @example
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless list_bucket_objects?(s3_client, 'doc-example-bucket')
def list_bucket_objects?(s3_client, bucket_name)
  puts "Accessing the bucket named '#{bucket_name}'..."
  objects = s3_client.list_objects_v2(
    bucket: bucket_name,
    max_keys: 50
  )

  if objects.count.positive?
    puts "The object keys in this bucket are (first 50 objects):"
    objects.contents.each do |object|
      puts object.key
    end
  else
```

```
    puts "No objects found in this bucket."
  end

  return true
rescue StandardError => e
  puts "Error while accessing the bucket named '#{bucket_name}': #{e.message}"
  return false
end

# Example usage:
def run_me
  region = "us-west-2"
  bucket_name = "BUCKET_NAME"
  s3_client = Aws::S3::Client.new(region: region)

  exit 1 unless list_bucket_objects?(s3_client, bucket_name)
end

run_me if $PROGRAM_NAME == __FILE__
```

Jika Anda tidak memiliki file AWS kredensial lokal, Anda masih dapat membuat `Aws::S3::Client` sumber daya dan menjalankan kode terhadap bucket dan objek Amazon S3. Permintaan yang dikirim menggunakan SDK versi 3 untuk Ruby bersifat anonim, tanpa tanda tangan default. Amazon S3 menampilkan pesan kesalahan jika Anda mengirim permintaan anonim untuk sumber daya yang tidak tersedia untuk umum.

Anda dapat menggunakan dan memperluas potongan kode sebelumnya untuk SDK untuk aplikasi Ruby, seperti dalam contoh yang lebih kuat berikut.

```
# Prerequisites:
#  - An existing Amazon S3 bucket.

require "aws-sdk-s3"

# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The bucket's name.
# @return [Boolean] true if all operations succeed; otherwise, false.
# @example
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless list_bucket_objects?(s3_client, 'doc-example-bucket')
def list_bucket_objects?(s3_client, bucket_name)
```

```
  puts "Accessing the bucket named '#{bucket_name}'..."
  objects = s3_client.list_objects_v2(
    bucket: bucket_name,
    max_keys: 50
  )

  if objects.count.positive?
    puts "The object keys in this bucket are (first 50 objects):"
    objects.contents.each do |object|
      puts object.key
    end
  else
    puts "No objects found in this bucket."
  end

  return true
rescue StandardError => e
  puts "Error while accessing the bucket named '#{bucket_name}': #{e.message}"
  return false
end

# Example usage:
def run_me
  region = "us-west-2"
  bucket_name = "BUCKET_NAME"
  s3_client = Aws::S3::Client.new(region: region)

  exit 1 unless list_bucket_objects?(s3_client, bucket_name)
end

run_me if $PROGRAM_NAME == __FILE__
```

Go

Example

Contoh berikut menggunakan AWS kredensil yang dimuat secara otomatis oleh SDK for Go dari file kredensial bersama.

```
package main

import (
```

```
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/s3"
)

// main uses the AWS SDK for Go V2 to create an Amazon Simple Storage Service
// (Amazon S3) client and list up to 10 buckets in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 s3Client := s3.NewFromConfig(sdkConfig)
 count := 10
 fmt.Printf("Let's list up to %v buckets for your account.\n", count)
 result, err := s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
 if err != nil {
  fmt.Printf("Couldn't list buckets for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Buckets) == 0 {
  fmt.Println("You don't have any buckets!")
 } else {
  if count > len(result.Buckets) {
   count = len(result.Buckets)
  }
  for _, bucket := range result.Buckets[:count] {
   fmt.Printf("\t%v\n", *bucket.Name)
  }
 }
}
```

Sumber daya terkait

- Mengembangkan dengan Amazon S3 menggunakan SDK AWS

- AWS SDK for PHP untuk Aws\S3\S3Client Class Amazon S3
- Dokumentasi AWS SDK for PHP

Membuat permintaan menggunakan kredensial sementara pengguna IAM

Pengguna IAM Akun AWS atau IAM dapat meminta kredensil keamanan sementara dan menggunakannya untuk mengirim permintaan yang diautentikasi ke Amazon S3. Bagian ini memberikan contoh cara menggunakan AWS SDK for Java, .NET, dan PHP untuk memperoleh kredensial keamanan sementara dan menggunakannya untuk mengautentikasi permintaan Anda ke Amazon S3.

Java

Pengguna IAM atau pengguna Akun AWS dapat meminta kredensil keamanan sementara (lihatMembuat permintaan) menggunakan AWS SDK for Java dan menggunakannya untuk mengakses Amazon S3. Kredensial ini berakhir setelah durasi sesi yang ditentukan.

Secara default, durasi sesi adalah satu jam. Jika Anda menggunakan kredensial pengguna IAM, Anda dapat menentukan durasi saat meminta kredensial keamanan sementara dari 15 menit hingga durasi sesi maksimum untuk peran tersebut. Untuk informasi selengkapnya tentang kredensial keamanan sementara, lihat Kredensial Keamanan Sementara dalam Panduan Pengguna IAM. Untuk informasi selengkapnya tentang membuat permintaan, lihat Membuat permintaan.

Untuk mendapatkan kredensial keamanan sementara dan mengakses Amazon S3

1. Buat instans dari kelas `AWSSecurityTokenService`. Untuk informasi tentang memberikan kredensial, lihat Mengembangkan dengan Amazon S3 menggunakan SDK AWS.
2. Ambil kredensial keamanan sementara untuk peran yang diinginkan dengan memanggil metode `assumeRole()` klien Security Token Service (STS).
3. Kemas kredensial keamanan sementara menjadi objek `BasicSessionCredentials`. Anda menggunakan objek ini untuk menyediakan kredensial keamanan sementara kepada klien Amazon S3.
4. Buat instans kelas `AmazonS3Client` dengan menggunakan kredensial keamanan sementara. Anda mengirim permintaan ke Amazon S3 menggunakan klien ini. Jika Anda mengirim permintaan menggunakan kredensial kedaluwarsa, Amazon S3 akan menampilkan pesan kesalahan.

> **Note**
>
> Jika Anda memperoleh kredensial keamanan sementara menggunakan kredensial keamanan Akun AWS Anda, kredensial sementara hanya berlaku selama satu jam. Anda dapat menentukan durasi sesi hanya jika Anda menggunakan kredensial pengguna IAM untuk meminta sesi.

Contoh berikut ini mencantumkan serangkaian kunci objek dalam bucket yang ditentukan. Contoh tersebut memperoleh kredensial keamanan sementara untuk suatu sesi dan menggunakannya untuk mengirim permintaan terautentikasi ke Amazon S3.

Jika Anda ingin menguji sampel menggunakan kredensial pengguna IAM, Anda harus membuat pengguna IAM di bawah Akun AWS Anda. Untuk informasi selengkapnya tentang cara membuat pengguna IAM, lihat Membuat Grup Pengguna dan Administrator IAM Pertama Anda dalam Panduan Pengguna IAM.

Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicSessionCredentials;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectListing;
import com.amazonaws.services.securitytoken.AWSSecurityTokenService;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClientBuilder;
import com.amazonaws.services.securitytoken.model.AssumeRoleRequest;
import com.amazonaws.services.securitytoken.model.AssumeRoleResult;
import com.amazonaws.services.securitytoken.model.Credentials;

public class MakingRequestsWithIAMTempCredentials {
    public static void main(String[] args) {
        String clientRegion = "*** Client region ***";
        String roleARN = "*** ARN for role to be assumed ***";
        String roleSessionName = "*** Role session name ***";
        String bucketName = "*** Bucket name ***";
```

```
        try {
            // Creating the STS client is part of your trusted code. It has
            // the security credentials you use to obtain temporary security
credentials.
            AWSSecurityTokenService stsClient =
AWSSecurityTokenServiceClientBuilder.standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();

            // Obtain credentials for the IAM role. Note that you cannot assume the
role of
            // an AWS root account;
            // Amazon S3 will deny access. You must use credentials for an IAM user
or an
            // IAM role.
            AssumeRoleRequest roleRequest = new AssumeRoleRequest()
                    .withRoleArn(roleARN)
                    .withRoleSessionName(roleSessionName);
            AssumeRoleResult roleResponse = stsClient.assumeRole(roleRequest);
            Credentials sessionCredentials = roleResponse.getCredentials();

            // Create a BasicSessionCredentials object that contains the credentials
you
            // just retrieved.
            BasicSessionCredentials awsCredentials = new BasicSessionCredentials(
                    sessionCredentials.getAccessKeyId(),
                    sessionCredentials.getSecretAccessKey(),
                    sessionCredentials.getSessionToken());

            // Provide temporary security credentials so that the Amazon S3 client
            // can send authenticated requests to Amazon S3. You create the client
            // using the sessionCredentials object.
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new
AWSStaticCredentialsProvider(awsCredentials))
                    .withRegion(clientRegion)
                    .build();

            // Verify that assuming the role worked and the permissions are set
correctly
            // by getting a set of object keys from the bucket.
            ObjectListing objects = s3Client.listObjects(bucketName);
```

```
            System.out.println("No. of Objects: " +
 objects.getObjectSummaries().size());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Pengguna IAM atau pengguna Akun AWS dapat meminta kredensil keamanan sementara menggunakan AWS SDK for .NET dan menggunakannya untuk mengakses Amazon S3. Kredensial ini berakhir setelah durasi sesi.

Secara default, durasi sesi adalah satu jam. Jika Anda menggunakan kredensial pengguna IAM, Anda dapat menentukan durasi saat meminta kredensial keamanan sementara dari 15 menit hingga durasi sesi maksimum untuk peran tersebut. Untuk informasi selengkapnya tentang kredensial keamanan sementara, lihat Kredensial Keamanan Sementara dalam Panduan Pengguna IAM. Untuk informasi selengkapnya tentang membuat permintaan, lihat Membuat permintaan.

Untuk mendapatkan kredensial keamanan sementara dan mengakses Amazon S3

1. Buat contoh AWS Security Token Service klien,`AmazonSecurityTokenServiceClient`. Untuk informasi tentang memberikan kredensial, lihat Mengembangkan dengan Amazon S3 menggunakan SDK AWS.
2. Mulai sesi dengan memanggil metode `GetSessionToken` klien STS yang Anda buat pada langkah sebelumnya. Anda memberikan informasi sesi untuk metode ini menggunakan objek `GetSessionTokenRequest`.

   Metode tersebut mengembalikan kredensial keamanan sementara Anda.

3. Kemas kredensial keamanan sementara dalam instans objek `SessionAWSCredentials`. Anda menggunakan objek ini untuk menyediakan kredensial keamanan sementara kepada klien Amazon S3.
4. Buat instans kelas `AmazonS3Client` dengan memasukkan kredensial keamanan sementara. Anda mengirim permintaan ke Amazon S3 menggunakan klien ini. Jika Anda mengirim permintaan menggunakan kredensial kedaluwarsa, Amazon S3 akan menampilkan pesan kesalahan.

> **Note**
>
> Jika Anda memperoleh kredensial keamanan sementara menggunakan kredensial keamanan Akun AWS Anda, kredensial sementara hanya berlaku selama satu jam. Anda dapat menentukan durasi sesi hanya jika Anda menggunakan kredensial pengguna IAM untuk meminta sesi.

Contoh C# berikut mencantumkan kunci objek dalam bucket yang ditentukan. Sebagai ilustrasi, contoh tersebut memperoleh kredensial keamanan sementara untuk sesi default satu jam dan menggunakannya untuk mengirim permintaan terautentikasi ke Amazon S3.

Jika Anda ingin menguji sampel menggunakan kredensial pengguna IAM, Anda harus membuat pengguna IAM di bawah Akun AWS Anda. Untuk informasi selengkapnya tentang cara membuat pengguna IAM, lihat Membuat Grup Pengguna dan Administrator IAM Pertama Anda dalam Panduan Pengguna IAM. Untuk informasi selengkapnya tentang membuat permintaan, lihat Membuat permintaan.

Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.Runtime;
using Amazon.S3;
using Amazon.S3.Model;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
```

```
namespace Amazon.DocSamples.S3
{
    class TempCredExplicitSessionStartTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;
        public static void Main()
        {
            ListObjectsAsync().Wait();
        }

        private static async Task ListObjectsAsync()
        {
            try
            {
                // Credentials use the default AWS SDK for .NET credential search
 chain.
                // On local development machines, this is your default profile.
                Console.WriteLine("Listing objects stored in a bucket");
                SessionAWSCredentials tempCredentials = await
 GetTemporaryCredentialsAsync();

                // Create a client by providing temporary security credentials.
                using (s3Client = new AmazonS3Client(tempCredentials, bucketRegion))
                {
                    var listObjectRequest = new ListObjectsRequest
                    {
                        BucketName = bucketName
                    };
                    // Send request to Amazon S3.
                    ListObjectsResponse response = await
 s3Client.ListObjectsAsync(listObjectRequest);
                    List<S3Object> objects = response.S3Objects;
                    Console.WriteLine("Object count = {0}", objects.Count);
                }
            }
            catch (AmazonS3Exception s3Exception)
            {
                Console.WriteLine(s3Exception.Message, s3Exception.InnerException);
            }
            catch (AmazonSecurityTokenServiceException stsException)
```

```
            {
                Console.WriteLine(stsException.Message,
 stsException.InnerException);
            }
        }

        private static async Task<SessionAWSCredentials>
 GetTemporaryCredentialsAsync()
        {
            using (var stsClient = new AmazonSecurityTokenServiceClient())
            {
                var getSessionTokenRequest = new GetSessionTokenRequest
                {
                    DurationSeconds = 7200 // seconds
                };

                GetSessionTokenResponse sessionTokenResponse =
                              await
 stsClient.GetSessionTokenAsync(getSessionTokenRequest);

                Credentials credentials = sessionTokenResponse.Credentials;

                var sessionCredentials =
                    new SessionAWSCredentials(credentials.AccessKeyId,
                                              credentials.SecretAccessKey,
                                              credentials.SessionToken);
                return sessionCredentials;
            }
        }
    }
}
```

PHP

Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

Pengguna IAM atau Akun AWS dapat meminta kredensil keamanan sementara menggunakan versi 3 dari file. AWS SDK for PHP Itu kemudian dapat menggunakan kredensial sementara untuk mengakses Amazon S3. Kredensial berakhir saat durasi sesi berakhir.

Secara default, durasi sesi adalah satu jam. Jika Anda menggunakan kredensial pengguna IAM, Anda dapat menentukan durasi saat meminta kredensial keamanan sementara dari 15 menit

hingga durasi sesi maksimum untuk peran tersebut. Untuk informasi selengkapnya tentang kredensial keamanan sementara, lihat Kredensial Keamanan Sementara dalam Panduan Pengguna IAM. Untuk informasi selengkapnya tentang membuat permintaan, lihat Membuat permintaan.

> **Note**
>
> Jika Anda memperoleh kredensial keamanan sementara menggunakan kredensial keamanan Akun AWS Anda, kredensial keamanan sementara hanya berlaku selama satu jam. Anda dapat menentukan durasi sesi hanya jika Anda menggunakan kredensial pengguna IAM untuk meminta sesi.

Example

Contoh PHP berikut ini mencantumkan kunci objek dalam bucket tertentu menggunakan kredensial keamanan sementara. Contoh tersebut memperoleh kredensial keamanan sementara untuk sesi satu jam default, dan menggunakannya untuk mengirim permintaan terautentikasi ke Amazon S3. Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

Jika Anda ingin menguji contoh menggunakan kredensial pengguna IAM, Anda harus membuat pengguna IAM di bawah Akun AWS Anda. Untuk informasi tentang cara membuat pengguna IAM, lihat Membuat Grup Pengguna dan Administrator IAM Pertama Anda dalam Panduan Pengguna IAM. Untuk contoh dari pengaturan durasi sesi saat menggunakan kredensial pengguna IAM untuk meminta sesi, lihat Membuat permintaan menggunakan kredensial sementara pengguna IAM .

```
 require 'vendor/autoload.php';

use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;

$bucket = '*** Your Bucket Name ***';

$sts = new StsClient([
    'version' => 'latest',
    'region' => 'us-east-1'
]);
```

```
$sessionToken = $sts->getSessionToken();

$s3 = new S3Client([
    'region' => 'us-east-1',
    'version' => 'latest',
    'credentials' => [
        'key'    => $sessionToken['Credentials']['AccessKeyId'],
        'secret' => $sessionToken['Credentials']['SecretAccessKey'],
        'token'  => $sessionToken['Credentials']['SessionToken']
    ]
]);

$result = $s3->listBuckets();

try {
    // Retrieve a paginator for listing objects.
    $objects = $s3->getPaginator('ListObjects', [
        'Bucket' => $bucket
    ]);

    echo "Keys retrieved!" . PHP_EOL;

    // List objects
    foreach ($objects as $object) {
        echo $object['Key'] . PHP_EOL;
    }
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Ruby

Pengguna IAM atau pengguna Akun AWS dapat meminta kredensil keamanan sementara menggunakan AWS SDK for Ruby dan menggunakannya untuk mengakses Amazon S3. Kredensial ini berakhir setelah durasi sesi.

Secara default, durasi sesi adalah satu jam. Jika Anda menggunakan kredensial pengguna IAM, Anda dapat menentukan durasi saat meminta kredensial keamanan sementara dari 15 menit hingga durasi sesi maksimum untuk peran tersebut. Untuk informasi selengkapnya tentang kredensial keamanan sementara, lihat Kredensial Keamanan Sementara dalam Panduan

Pengguna IAM. Untuk informasi selengkapnya tentang membuat permintaan, lihat Membuat permintaan.

> **Note**
>
> Jika Anda memperoleh kredensial keamanan sementara menggunakan kredensial keamanan Akun AWS Anda, kredensial keamanan sementara hanya berlaku selama satu jam. Anda dapat menentukan durasi sesi hanya jika Anda menggunakan kredensial pengguna IAM untuk meminta sesi.

Contoh Ruby berikut membuat pengguna sementara untuk mencantumkan item dalam bucket tertentu selama satu jam. Untuk menggunakan contoh ini, Anda harus memiliki AWS kredensil yang memiliki izin yang diperlukan untuk membuat klien new AWS Security Token Service (AWS STS), dan mencantumkan bucket Amazon S3.

```
# Prerequisites:
# - A user in AWS Identity and Access Management (IAM). This user must
#   be able to assume the following IAM role. You must run this code example
#   within the context of this user.
# - An existing role in IAM that allows all of the Amazon S3 actions for all of the
#   resources in this code example. This role must also trust the preceding IAM
 user.
# - An existing S3 bucket.

require "aws-sdk-core"
require "aws-sdk-s3"
require "aws-sdk-iam"

# Checks whether a user exists in IAM.
#
# @param iam [Aws::IAM::Client] An initialized IAM client.
# @param user_name [String] The user's name.
# @return [Boolean] true if the user exists; otherwise, false.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-west-2')
#   exit 1 unless user_exists?(iam_client, 'my-user')
def user_exists?(iam_client, user_name)
  response = iam_client.get_user(user_name: user_name)
  return true if response.user.user_name
```

```
rescue Aws::IAM::Errors::NoSuchEntity
  # User doesn't exist.
rescue StandardError => e
  puts "Error while determining whether the user " \
    "'#{user_name}' exists: #{e.message}"
end

# Creates a user in IAM.
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param user_name [String] The user's name.
# @return [AWS:IAM::Types::User] The new user.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-west-2')
#   user = create_user(iam_client, 'my-user')
#   exit 1 unless user.user_name
def create_user(iam_client, user_name)
  response = iam_client.create_user(user_name: user_name)
  return response.user
rescue StandardError => e
  puts "Error while creating the user '#{user_name}': #{e.message}"
end

# Gets a user in IAM.
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param user_name [String] The user's name.
# @return [AWS:IAM::Types::User] The existing user.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-west-2')
#   user = get_user(iam_client, 'my-user')
#   exit 1 unless user.user_name
def get_user(iam_client, user_name)
  response = iam_client.get_user(user_name: user_name)
  return response.user
rescue StandardError => e
  puts "Error while getting the user '#{user_name}': #{e.message}"
end

# Checks whether a role exists in IAM.
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param role_name [String] The role's name.
# @return [Boolean] true if the role exists; otherwise, false.
```

```
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-west-2')
#   exit 1 unless role_exists?(iam_client, 'my-role')
def role_exists?(iam_client, role_name)
  response = iam_client.get_role(role_name: role_name)
  return true if response.role.role_name
rescue StandardError => e
  puts "Error while determining whether the role " \
    "'#{role_name}' exists: #{e.message}"
end

# Gets credentials for a role in IAM.
#
# @param sts_client [Aws::STS::Client] An initialized AWS STS client.
# @param role_arn [String] The role's Amazon Resource Name (ARN).
# @param role_session_name [String] A name for this role's session.
# @param duration_seconds [Integer] The number of seconds this session is valid.
# @return [AWS::AssumeRoleCredentials] The credentials.
# @example
#   sts_client = Aws::STS::Client.new(region: 'us-west-2')
#   credentials = get_credentials(
#     sts_client,
#     'arn:aws:iam::123456789012:role/AmazonS3ReadOnly',
#     'ReadAmazonS3Bucket',
#     3600
#   )
#   exit 1 if credentials.nil?
def get_credentials(sts_client, role_arn, role_session_name, duration_seconds)
  Aws::AssumeRoleCredentials.new(
    client: sts_client,
    role_arn: role_arn,
    role_session_name: role_session_name,
    duration_seconds: duration_seconds
  )
rescue StandardError => e
  puts "Error while getting credentials: #{e.message}"
end

# Checks whether a bucket exists in Amazon S3.
#
# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The name of the bucket.
# @return [Boolean] true if the bucket exists; otherwise, false.
# @example
```

```
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless bucket_exists?(s3_client, 'doc-example-bucket')
def bucket_exists?(s3_client, bucket_name)
  response = s3_client.list_buckets
  response.buckets.each do |bucket|
    return true if bucket.name == bucket_name
  end
rescue StandardError => e
  puts "Error while checking whether the bucket '#{bucket_name}' " \
    "exists: #{e.message}"
end

# Lists the keys and ETags for the objects in an Amazon S3 bucket.
#
# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The bucket's name.
# @return [Boolean] true if the objects were listed; otherwise, false.
# @example
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless list_objects_in_bucket?(s3_client, 'doc-example-bucket')
def list_objects_in_bucket?(s3_client, bucket_name)
  puts "Accessing the contents of the bucket named '#{bucket_name}'..."
  response = s3_client.list_objects_v2(
    bucket: bucket_name,
    max_keys: 50
  )

  if response.count.positive?
    puts "Contents of the bucket named '#{bucket_name}' (first 50 objects):"
    puts "Name => ETag"
    response.contents.each do |obj|
      puts "#{obj.key} => #{obj.etag}"
    end
  else
    puts "No objects in the bucket named '#{bucket_name}'."
  end
  return true
rescue StandardError => e
  puts "Error while accessing the bucket named '#{bucket_name}': #{e.message}"
end
```

Sumber daya terkait

- Mengembangkan dengan Amazon S3 menggunakan SDK AWS
- AWS SDK for PHP untuk Amazon S3 Aws\ S3\ S3Kelas Klien
- Dokumentasi AWS SDK for PHP

Membuat permintaan menggunakan kredensial sementara pengguna gabungan

Anda dapat meminta kredensi keamanan sementara dan memberikannya kepada pengguna atau aplikasi federasi Anda yang perlu mengakses sumber daya Anda. AWS Bagian ini memberikan contoh bagaimana Anda dapat menggunakan AWS SDK untuk mendapatkan kredenal keamanan sementara bagi pengguna atau aplikasi federasi Anda dan mengirim permintaan yang diautentikasi ke Amazon S3 menggunakan kredensi tersebut. Untuk daftar AWS SDK yang tersedia, lihat Contoh Kode dan Pustaka.

> **Note**
>
> Baik pengguna IAM Akun AWS maupun IAM dapat meminta kredensi keamanan sementara untuk pengguna federasi. Namun, untuk keamanan tambahan, hanya pengguna IAM dengan izin yang diperlukan yang harus meminta kredensial sementara ini untuk memastikan bahwa pengguna gabungan mendapatkan paling banyak izin dari pengguna IAM yang meminta. Dalam beberapa aplikasi, Anda mungkin menganggapnya sesuai untuk membuat pengguna IAM dengan izin tertentu semata-mata untuk tujuan memberikan kredensial keamanan sementara kepada pengguna gabungan dan aplikasi Anda.

Java

Anda dapat memberikan kredensi keamanan sementara untuk pengguna dan aplikasi federasi Anda sehingga mereka dapat mengirim permintaan yang diautentikasi untuk mengakses sumber daya Anda. AWS Saat meminta kredensial sementara ini, Anda harus memberikan nama pengguna dan kebijakan IAM yang menjelaskan izin sumber daya yang ingin Anda berikan. Secara default, durasi sesi adalah satu jam. Anda dapat secara eksplisit mengatur nilai durasi yang berbeda ketika meminta kredensial keamanan sementara untuk pengguna gabungan dan aplikasi.

> **Note**
>
> Untuk keamanan tambahan saat meminta kredensial keamanan sementara untuk pengguna gabungan dan aplikasi, kami menyarankan agar Anda menggunakan pengguna IAM khusus hanya dengan izin akses yang diperlukan. Pengguna sementara yang Anda buat tidak akan pernah mendapatkan izin lebih dari pengguna IAM yang meminta kredensial keamanan sementara. Untuk informasi selengkapnya, lihat FAQ AWS Identity and Access Management.

Untuk memberikan kredensial keamanan dan mengirimkan permintaan terautentikasi untuk mengakses sumber daya, lakukan hal berikut:

- Buat instans dari kelas `AWSSecurityTokenServiceClient`.
- Mulai sesi dengan memanggil metode `getFederationToken()` klien Security Token Service (STS). Berikan informasi sesi, termasuk nama pengguna dan kebijakan IAM, yang ingin Anda lampirkan ke kredensial sementara. Anda dapat memberikan durasi sesi opsional. Metode tersebut mengembalikan kredensial keamanan sementara Anda.
- Kemas kredensial keamanan sementara dalam instans objek `BasicSessionCredentials`. Anda menggunakan objek ini untuk menyediakan kredensial keamanan sementara kepada klien Amazon S3.
- Buat instans kelas `AmazonS3Client` dengan menggunakan kredensial keamanan sementara. Anda mengirim permintaan ke Amazon S3 menggunakan klien ini. Jika Anda mengirim permintaan menggunakan kredensial kedaluwarsa, Amazon S3 akan menampilkan pesan kesalahan.

Example

Contoh ini mencantumkan kunci dalam bucket S3 yang ditentukan. Dalam contoh, Anda memperoleh kredensial keamanan sementara untuk sesi dua jam bagi pengguna gabungan Anda dan menggunakan kredensial untuk mengirim permintaan yang diautentikasi ke Amazon S3. Untuk menjalankan contoh, Anda perlu membuat pengguna IAM dengan kebijakan terlampir yang memungkinkan pengguna untuk meminta kredensi keamanan sementara dan daftar sumber daya Anda. AWS Kebijakan berikut mencapai hal ini:

```
{
  "Statement":[{
      "Action":["s3:ListBucket",
        "sts:GetFederationToken*"
      ],
      "Effect":"Allow",
      "Resource":"*"
    }
  ]
}
```

Untuk informasi selengkapnya tentang cara membuat pengguna IAM, lihat Membuat Grup Pengguna dan Administrator IAM Pertama Anda dalam Panduan Pengguna IAM.

Setelah membuat pengguna IAM dan melampirkan kebijakan sebelumnya, Anda dapat menjalankan contoh berikut. Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicSessionCredentials;
import com.amazonaws.auth.policy.Policy;
import com.amazonaws.auth.policy.Resource;
import com.amazonaws.auth.policy.Statement;
import com.amazonaws.auth.policy.Statement.Effect;
import com.amazonaws.auth.policy.actions.S3Actions;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectListing;
import com.amazonaws.services.securitytoken.AWSSecurityTokenService;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClientBuilder;
import com.amazonaws.services.securitytoken.model.Credentials;
import com.amazonaws.services.securitytoken.model.GetFederationTokenRequest;
import com.amazonaws.services.securitytoken.model.GetFederationTokenResult;

import java.io.IOException;

public class MakingRequestsWithFederatedTempCredentials {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String bucketName = "*** Specify bucket name ***";
        String federatedUser = "*** Federated user name ***";
        String resourceARN = "arn:aws:s3:::" + bucketName;

        try {
            AWSSecurityTokenService stsClient = AWSSecurityTokenServiceClientBuilder
                    .standard()
                    .withCredentials(new ProfileCredentialsProvider())
                    .withRegion(clientRegion)
                    .build();
```

```
            GetFederationTokenRequest getFederationTokenRequest = new
GetFederationTokenRequest();
            getFederationTokenRequest.setDurationSeconds(7200);
            getFederationTokenRequest.setName(federatedUser);

            // Define the policy and add it to the request.
            Policy policy = new Policy();
            policy.withStatements(new Statement(Effect.Allow)
                    .withActions(S3Actions.ListObjects)
                    .withResources(new Resource(resourceARN)));
            getFederationTokenRequest.setPolicy(policy.toJson());

            // Get the temporary security credentials.
            GetFederationTokenResult federationTokenResult =
stsClient.getFederationToken(getFederationTokenRequest);
            Credentials sessionCredentials = federationTokenResult.getCredentials();

            // Package the session credentials as a BasicSessionCredentials
            // object for an Amazon S3 client object to use.
            BasicSessionCredentials basicSessionCredentials = new
BasicSessionCredentials(
                    sessionCredentials.getAccessKeyId(),
                    sessionCredentials.getSecretAccessKey(),
                    sessionCredentials.getSessionToken());
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withCredentials(new
AWSStaticCredentialsProvider(basicSessionCredentials))
                    .withRegion(clientRegion)
                    .build();

            // To verify that the client works, send a listObjects request using
            // the temporary security credentials.
            ObjectListing objects = s3Client.listObjects(bucketName);
            System.out.println("No. of Objects = " +
objects.getObjectSummaries().size());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
```

```
    }
}
```

.NET

Anda dapat memberikan kredensi keamanan sementara untuk pengguna dan aplikasi federasi Anda sehingga mereka dapat mengirim permintaan yang diautentikasi untuk mengakses sumber daya Anda. AWS Saat meminta kredensial sementara ini, Anda harus memberikan nama pengguna dan kebijakan IAM yang menjelaskan izin sumber daya yang ingin Anda berikan. Secara default, durasi sesi adalah satu jam. Anda dapat secara eksplisit mengatur nilai durasi yang berbeda ketika meminta kredensial keamanan sementara untuk pengguna gabungan dan aplikasi. Untuk informasi tentang mengirim permintaan terautentikasi, lihat Membuat permintaan.

> **Note**
>
> Saat meminta kredensial keamanan sementara untuk pengguna gabungan dan aplikasi, untuk keamanan tambahan, kami sarankan Anda menggunakan pengguna IAM khusus hanya dengan izin akses yang diperlukan. Pengguna sementara yang Anda buat tidak akan pernah mendapatkan izin lebih dari pengguna IAM yang meminta kredensial keamanan sementara. Untuk informasi selengkapnya, lihat FAQ AWS Identity and Access Management.

Anda melakukan hal berikut:

- Buat instance AWS Security Token Service klien, `AmazonSecurityTokenServiceClient` kelas.
- Mulai sesi dengan memanggil metode `GetFederationToken` klien STS. Anda perlu memberikan informasi sesi, termasuk nama pengguna dan kebijakan IAM yang ingin Anda lampirkan ke kredensial sementara. Atau, Anda dapat memberikan durasi sesi. Metode tersebut mengembalikan kredensial keamanan sementara Anda.
- Kemas kredensial keamanan sementara dalam instans objek `SessionAWSCredentials`. Anda menggunakan objek ini untuk menyediakan kredensial keamanan sementara kepada klien Amazon S3.
- Buat instans kelas `AmazonS3Client` dengan menyerahkan kredensial keamanan sementara. Anda menggunakan klien ini untuk mengirim permintaan ke Amazon S3. Jika Anda mengirim

permintaan menggunakan kredensial kedaluwarsa, Amazon S3 akan menampilkan pesan kesalahan.

Example

Contoh C# berikut mencantumkan kunci dalam bucket yang ditentukan. Dalam contoh, Anda memperoleh kredensial keamanan sementara untuk sesi dua jam bagi pengguna gabungan (Pengguna 1) Anda dan menggunakan kredensial untuk mengirim permintaan yang diautentikasi ke Amazon S3.

- Untuk latihan ini, Anda membuat pengguna IAM dengan izin minimal. Dengan menggunakan kredensial pengguna IAM ini, Anda meminta kredensial sementara untuk orang lain. Contoh ini hanya mencantumkan objek dalam bucket tertentu. Buat pengguna IAM dengan kebijakan berikut terlampir:

  ```
  {
    "Statement":[{
        "Action":["s3:ListBucket",
          "sts:GetFederationToken*"
        ],
        "Effect":"Allow",
        "Resource":"*"
      }
    ]
  }
  ```

  Kebijakan ini memungkinkan pengguna IAM untuk meminta kredenial keamanan sementara dan izin akses hanya untuk mencantumkan sumber daya Anda. AWS Untuk informasi selengkapnya tentang cara membuat pengguna IAM, lihat Membuat Grup Pengguna dan Administrator IAM Anda dalam Panduan Pengguna IAM.
- Gunakan kredensial keamanan pengguna IAM untuk menguji contoh berikut. Contoh tersebut mengirimkan permintaan terautentikasi ke Amazon S3 menggunakan kredensial keamanan sementara. Contoh tersebut menentukan kebijakan berikut ketika meminta kredensial keamanan sementara untuk pengguna gabungan (Pengguna1), yang membatasi akses untuk mencantumkan objek dalam bucket tertentu (`YourBucketName`). Anda harus memperbarui kebijakan dan memberikan nama bucket Anda sendiri.

  ```
  {
    "Statement":[
  ```

```
    {
      "Sid":"1",
      "Action":["s3:ListBucket"],
      "Effect":"Allow",
      "Resource":"arn:aws:s3:::YourBucketName"
    }
  ]
}
```

- Example

  Perbarui sampel berikut dan berikan nama bucket yang Anda tentukan dalam kebijakan akses pengguna gabungan sebelumnya. Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.Runtime;
using Amazon.S3;
using Amazon.S3.Model;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;
using System;
using System.Collections.Generic;
using System.Threading.Tasks;

namespace Amazon.DocSamples.S3
{
    class TempFederatedCredentialsTest
    {
        private const string bucketName = "*** bucket name ***";
        // Specify your bucket region (an example region is shown).
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 client;

        public static void Main()
        {
            ListObjectsAsync().Wait();
        }

        private static async Task ListObjectsAsync()
        {
            try
```

```
            {
                Console.WriteLine("Listing objects stored in a bucket");
                // Credentials use the default AWS SDK for .NET credential search
chain.
                // On local development machines, this is your default profile.
                SessionAWSCredentials tempCredentials =
                    await GetTemporaryFederatedCredentialsAsync();

                // Create a client by providing temporary security credentials.
                using (client = new AmazonS3Client(bucketRegion))
                {
                    ListObjectsRequest listObjectRequest = new
ListObjectsRequest();
                    listObjectRequest.BucketName = bucketName;

                    ListObjectsResponse response = await
client.ListObjectsAsync(listObjectRequest);
                    List<S3Object> objects = response.S3Objects;
                    Console.WriteLine("Object count = {0}", objects.Count);

                    Console.WriteLine("Press any key to continue...");
                    Console.ReadKey();
                }
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered ***. Message:'{0}' when
writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}'
when writing an object", e.Message);
            }
        }

        private static async Task<SessionAWSCredentials>
GetTemporaryFederatedCredentialsAsync()
        {
            AmazonSecurityTokenServiceConfig config = new
AmazonSecurityTokenServiceConfig();
            AmazonSecurityTokenServiceClient stsClient =
                new AmazonSecurityTokenServiceClient(
                                                 config);
```

```
            GetFederationTokenRequest federationTokenRequest =
                                    new GetFederationTokenRequest();
            federationTokenRequest.DurationSeconds = 7200;
            federationTokenRequest.Name = "User1";
            federationTokenRequest.Policy = @"{
               ""Statement"":
               [
                 {
                   ""Sid"":""Stmt1311212314284"",
                   ""Action"":[""s3:ListBucket""],
                   ""Effect"":""Allow"",
                   ""Resource"":""arn:aws:s3:::" + bucketName + @"""
                  }
               ]
             }
            ";

            GetFederationTokenResponse federationTokenResponse =
                        await
 stsClient.GetFederationTokenAsync(federationTokenRequest);
            Credentials credentials = federationTokenResponse.Credentials;

            SessionAWSCredentials sessionCredentials =
                new SessionAWSCredentials(credentials.AccessKeyId,
                                          credentials.SecretAccessKey,
                                          credentials.SessionToken);
            return sessionCredentials;
        }
    }
}
```

PHP

Topik ini menjelaskan cara menggunakan kelas dari versi 3 AWS SDK for PHP untuk meminta kredenal keamanan sementara untuk pengguna dan aplikasi federasi dan menggunakannya untuk mengakses sumber daya yang disimpan di Amazon S3. Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

Anda dapat memberikan kredensi keamanan sementara kepada pengguna dan aplikasi federasi Anda sehingga mereka dapat mengirim permintaan yang diautentikasi untuk mengakses sumber daya Anda. AWS Saat meminta kredensial sementara ini, Anda harus memberikan nama

pengguna dan kebijakan IAM yang menjelaskan izin sumber daya yang ingin Anda berikan. Kredensial ini berakhir saat durasi sesi berakhir. Secara default, durasi sesi adalah satu jam. Anda dapat secara eksplisit mengatur nilai yang berbeda selama durasi saat meminta kredensial keamanan sementara untuk pengguna gabungan dan aplikasi. Untuk informasi selengkapnya tentang kredensial keamanan sementara, lihat Kredensial Keamanan Sementara dalam Panduan Pengguna IAM. Untuk informasi tentang penyediaan kredensial keamanan sementara bagi pengguna gabungan dan aplikasi Anda, lihat Membuat permintaan.

Untuk keamanan tambahan saat meminta kredensial keamanan sementara untuk pengguna gabungan dan aplikasi, kami menyarankan untuk menggunakan pengguna IAM khusus hanya dengan izin akses yang diperlukan. Pengguna sementara yang Anda buat tidak akan pernah mendapatkan izin lebih dari pengguna IAM yang meminta kredensial keamanan sementara. Untuk informasi tentang federasi identitas, lihat FAQ AWS Identity and Access Management.

Untuk informasi selengkapnya tentang AWS SDK for Ruby API, AWS buka SDK for Ruby - Versi 2.

Example

Contoh PHP berikut ini mencantumkan kunci dalam bucket yang ditentukan. Dalam contoh ini, Anda memperoleh kredensial keamanan sementara untuk satu sesi jam bagi pengguna gabungan Anda (Pengguna1). Kemudian Anda menggunakan kredensial keamanan sementara untuk mengirim permintaan terautentikasi ke Amazon S3.

Untuk keamanan tambahan saat meminta kredensial sementara untuk orang lain, Anda menggunakan kredensial keamanan pengguna IAM yang memiliki izin untuk meminta kredensial keamanan sementara. Untuk memastikan bahwa pengguna IAM hanya memberikan izin spesifik aplikasi minimum kepada pengguna gabungan, Anda juga dapat membatasi izin akses pengguna IAM ini. Contoh ini hanya mencantumkan objek dalam bucket tertentu. Buat pengguna IAM dengan kebijakan berikut terlampir:

```
{
  "Statement":[{
      "Action":["s3:ListBucket",
        "sts:GetFederationToken*"
      ],
      "Effect":"Allow",
      "Resource":"*"
    }
  ]
```

```
}
```

Kebijakan ini memungkinkan pengguna IAM untuk meminta kredenial keamanan sementara dan izin akses hanya untuk mencantumkan sumber daya Anda. AWS Untuk informasi selengkapnya tentang cara membuat pengguna IAM, lihat Membuat Grup Pengguna dan Administrator IAM Pertama Anda dalam Panduan Pengguna IAM.

Sekarang Anda dapat menggunakan kredensial keamanan pengguna IAM untuk menguji contoh berikut. Contoh tersebut mengirimkan permintaan terautentikasi ke Amazon S3 menggunakan kredensial keamanan sementara. Ketika meminta kredensial keamanan sementara untuk pengguna gabungan (Pengguna1), contoh tersebut menentukan kebijakan berikut, yang membatasi akses ke objek daftar dalam bucket tertentu. Perbarui kebijakan dengan nama bucket Anda.

```
{
  "Statement":[
    {
      "Sid":"1",
      "Action":["s3:ListBucket"],
      "Effect":"Allow",
      "Resource":"arn:aws:s3:::YourBucketName"
    }
  ]
}
```

Dalam contoh berikut, ketika menentukan sumber daya kebijakan, ganti `YourBucketName` dengan nama bucket Anda.:

```
 require 'vendor/autoload.php';

use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;

$bucket = '*** Your Bucket Name ***';

// In real applications, the following code is part of your trusted code. It has
// the security credentials that you use to obtain temporary security credentials.
$sts = new StsClient([
    'version' => 'latest',
    'region' => 'us-east-1'
```

```
]);

// Fetch the federated credentials.
$sessionToken = $sts->getFederationToken([
    'Name'              => 'User1',
    'DurationSeconds'    => '3600',
    'Policy'            => json_encode([
        'Statement' => [
            'Sid'              => 'randomstatementid' . time(),
            'Action'           => ['s3:ListBucket'],
            'Effect'           => 'Allow',
            'Resource'         => 'arn:aws:s3:::' . $bucket
        ]
    ])
]);

// The following will be part of your less trusted code. You provide temporary
// security credentials so the code can send authenticated requests to Amazon S3.

$s3 = new S3Client([
    'region' => 'us-east-1',
    'version' => 'latest',
    'credentials' => [
        'key' => $sessionToken['Credentials']['AccessKeyId'],
        'secret' => $sessionToken['Credentials']['SecretAccessKey'],
        'token' => $sessionToken['Credentials']['SessionToken']
    ]
]);

try {
    $result = $s3->listObjects([
        'Bucket' => $bucket
    ]);
} catch (S3Exception $e) {
    echo $e->getMessage() . PHP_EOL;
}
```

Ruby

Anda dapat memberikan kredensi keamanan sementara untuk pengguna dan aplikasi federasi Anda sehingga mereka dapat mengirim permintaan yang diautentikasi untuk mengakses sumber daya Anda. AWS Saat meminta kredensial sementara dari layanan IAM, Anda harus memberikan

nama pengguna dan kebijakan IAM yang menjelaskan izin sumber daya yang ingin Anda berikan. Secara default, durasi sesi adalah satu jam. Namun, jika Anda meminta kredensial sementara menggunakan kredensial pengguna IAM, Anda dapat secara eksplisit mengatur nilai durasi yang berbeda ketika meminta kredensial keamanan sementara untuk pengguna gabungan dan aplikasi. Untuk informasi tentang kredensial keamanan sementara untuk pengguna dan aplikasi gabungan Anda, lihat Membuat permintaan.

> **Note**
>
> Untuk keamanan tambahan ketika Anda meminta kredensial keamanan sementara untuk pengguna gabungan dan aplikasi, Anda mungkin ingin menggunakan pengguna IAM khusus hanya dengan izin akses yang diperlukan. Pengguna sementara yang Anda buat tidak akan pernah mendapatkan izin lebih dari pengguna IAM yang meminta kredensial keamanan sementara. Untuk informasi selengkapnya, lihat FAQ AWS Identity and Access Management.

Example

Contoh kode Ruby berikut memungkinkan pengguna gabungan dengan set izin terbatas untuk mencantumkan kunci dalam bucket tertentu.

```
# Prerequisites:
#  - An existing Amazon S3 bucket.

require "aws-sdk-s3"
require "aws-sdk-iam"
require "json"

# Checks to see whether a user exists in IAM; otherwise,
# creates the user.
#
# @param iam [Aws::IAM::Client] An initialized IAM client.
# @param user_name [String] The user's name.
# @return [Aws::IAM::Types::User] The existing or new user.
# @example
#   iam = Aws::IAM::Client.new(region: 'us-west-2')
#   user = get_user(iam, 'my-user')
#   exit 1 unless user.user_name
#   puts "User's name: #{user.user_name}"
def get_user(iam, user_name)
```

```
  puts "Checking for a user with the name '#{user_name}'..."
  response = iam.get_user(user_name: user_name)
  puts "A user with the name '#{user_name}' already exists."
  return response.user
# If the user doesn't exist, create them.
rescue Aws::IAM::Errors::NoSuchEntity
  puts "A user with the name '#{user_name}' doesn't exist. Creating this user..."
  response = iam.create_user(user_name: user_name)
  iam.wait_until(:user_exists, user_name: user_name)
  puts "Created user with the name '#{user_name}'."
  return response.user
rescue StandardError => e
  puts "Error while accessing or creating the user named '#{user_name}':
 #{e.message}"
end

# Gets temporary AWS credentials for an IAM user with the specified permissions.
#
# @param sts [Aws::STS::Client] An initialized AWS STS client.
# @param duration_seconds [Integer] The number of seconds for valid credentials.
# @param user_name [String] The user's name.
# @param policy [Hash] The access policy.
# @return [Aws::STS::Types::Credentials] AWS credentials for API authentication.
# @example
#   sts = Aws::STS::Client.new(region: 'us-west-2')
#   credentials = get_temporary_credentials(sts, duration_seconds, user_name,
#     {
#       'Version' => '2012-10-17',
#       'Statement' => [
#         'Sid' => 'Stmt1',
#         'Effect' => 'Allow',
#         'Action' => 's3:ListBucket',
#         'Resource' => 'arn:aws:s3:::doc-example-bucket'
#       ]
#     }
#   )
#   exit 1 unless credentials.access_key_id
#   puts "Access key ID: #{credentials.access_key_id}"
def get_temporary_credentials(sts, duration_seconds, user_name, policy)
  response = sts.get_federation_token(
    duration_seconds: duration_seconds,
    name: user_name,
    policy: policy.to_json
  )
```

```
  return response.credentials
rescue StandardError => e
  puts "Error while getting federation token: #{e.message}"
end

# Lists the keys and ETags for the objects in an Amazon S3 bucket.
#
# @param s3_client [Aws::S3::Client] An initialized Amazon S3 client.
# @param bucket_name [String] The bucket's name.
# @return [Boolean] true if the objects were listed; otherwise, false.
# @example
#   s3_client = Aws::S3::Client.new(region: 'us-west-2')
#   exit 1 unless list_objects_in_bucket?(s3_client, 'doc-example-bucket')
def list_objects_in_bucket?(s3_client, bucket_name)
  puts "Accessing the contents of the bucket named '#{bucket_name}'..."
  response = s3_client.list_objects_v2(
    bucket: bucket_name,
    max_keys: 50
  )

  if response.count.positive?
    puts "Contents of the bucket named '#{bucket_name}' (first 50 objects):"
    puts "Name => ETag"
    response.contents.each do |obj|
      puts "#{obj.key} => #{obj.etag}"
    end
  else
    puts "No objects in the bucket named '#{bucket_name}'."
  end
  return true
rescue StandardError => e
  puts "Error while accessing the bucket named '#{bucket_name}': #{e.message}"
end

# Example usage:
def run_me
  region = "us-west-2"
  user_name = "my-user"
  bucket_name = "doc-example-bucket"

  iam = Aws::IAM::Client.new(region: region)
  user = get_user(iam, user_name)

  exit 1 unless user.user_name
```

```
  puts "User's name: #{user.user_name}"
  sts = Aws::STS::Client.new(region: region)
  credentials = get_temporary_credentials(sts, 3600, user_name,
    {
      "Version" => "2012-10-17",
      "Statement" => [
        "Sid" => "Stmt1",
        "Effect" => "Allow",
        "Action" => "s3:ListBucket",
        "Resource" => "arn:aws:s3:::#{bucket_name}"
      ]
    }
  )

  exit 1 unless credentials.access_key_id

  puts "Access key ID: #{credentials.access_key_id}"
  s3_client = Aws::S3::Client.new(region: region, credentials: credentials)

  exit 1 unless list_objects_in_bucket?(s3_client, bucket_name)
end

run_me if $PROGRAM_NAME == __FILE__
```

Sumber daya terkait

- Mengembangkan dengan Amazon S3 menggunakan SDK AWS
- AWS SDK for PHP untuk Amazon S3 Aws\ S3\ S3Kelas Klien
- Dokumentasi AWS SDK for PHP

Membuat permintaan menggunakan API REST

Bagian ini berisi informasi tentang cara membuat permintaan ke titik akhir Amazon S3 dengan menggunakan API REST. Untuk daftar endpoint Amazon S3, lihat Wilayah dan Titik Akhir di bagian Referensi Umum AWS.

Membangun hostname S3 untuk permintaan API REST

Titik akhir Amazon S3 mengikuti struktur yang ditunjukkan di bawah ini:

```
s3.Region.amazonaws.com
```

Titik akhir titik akses Amazon S3 dan titik akhir dual-stack juga mengikuti struktur standar:

- Titik akses Amazon S3 -`s3-accesspoint.Region.amazonaws.com`
- Dual-stack - `s3.dualstack.Region.amazonaws.com`

Untuk daftar lengkap Wilayah dan titik akhir Amazon S3, lihat Titik akhir dan kuota Amazon S3 dalam Referensi Umum Amazon Web Services.

Permintaan cara hosting virtual dan cara jalur

Saat membuat permintaan dengan menggunakan REST API, Anda dapat menggunakan URI cara hosting virtual atau cara jalur untuk titik akhir Amazon S3. Untuk informasi selengkapnya, lihat Bucket dengan hosting virtual.

Example Permintaan Cara Hosting Virtual

Berikut ini adalah contoh permintaan cara hosting virtual untuk menghapus `puppy.jpg` berkas dari bucket yang diberi nama `examplebucket` di Wilayah Barat AS (Oregon). Untuk informasi lebih lanjut tentang permintaan cara hosting virtual, lihat Permintaan bergaya host virtual.

```
DELETE /puppy.jpg HTTP/1.1
Host: examplebucket.s3.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Example Permintaan cara jalur

Berikut ini adalah contoh versi cara jalur dari permintaan yang sama.

```
DELETE /examplebucket/puppy.jpg HTTP/1.1
Host: s3.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Saat ini, Amazon S3 mendukung akses URL bergaya virtual-host dan styleWilayah AWS. Namun, URL gaya jalur akan dihentikan di masa future. Untuk informasi lebih lanjut, lihat catatan penting berikut.

Untuk informasi lebih lanjut tentang permintaan dan cara jalur, lihat Permintaan gaya jalur.

> **Important**
>
> Pembaruan (23 September 2020) — Untuk memastikan bahwa pelanggan memiliki waktu yang mereka butuhkan untuk beralih ke URL bergaya virtual-host dan style. Untuk informasi selengkapnya, lihat Rencana Penghentian Jalur Amazon S3 – Kisah Selanjutnya dalam Blog Berita AWS.

Membuat permintaan ke titik akhir dual-stack dengan menggunakan API REST

Saat menggunakan API REST, Anda dapat mengakses langsung titik akhir dual-stack dengan menggunakan nama titik akhir (URI) cara hosting virtual dan cara jalur. Semua nama titik akhir dual-stack Amazon S3 mencakup wilayah dalam nama tersebut. Tidak seperti titik akhir standar IPv4 saja, baik titik akhir cara hosting virtual dan cara jalur menggunakan nama titik akhir spesifik-wilayah.

Example Permintaan titik akhir dual-stack cara hosting virtual

Anda dapat menggunakan titik akhir cara hosting virtual dalam permintaan REST seperti yang ditunjukkan dalam contoh berikut yang mengambil objek `puppy.jpg` dari bucket yang diberi nama `examplebucket` di Wilayah Barat AS (Oregon).

```
GET /puppy.jpg HTTP/1.1
Host: examplebucket.s3.dualstack.us-west-2.amazonaws.com
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Example Permintaan titik akhir dual-stack cara jalur

Atau Anda dapat menggunakan titik akhir cara jalur dalam permintaan Anda seperti ditunjukkan dalam contoh berikut.

```
GET /examplebucket/puppy.jpg HTTP/1.1
Host: s3.dualstack.us-west-2.amazonaws.com
```

```
Date: Mon, 11 Apr 2016 12:00:00 GMT
x-amz-date: Mon, 11 Apr 2016 12:00:00 GMT
Authorization: authorization string
```

Untuk informasi lebih lanjut tentang titik akhir dual-stack, lihat Menggunakan titik akhir tumpukan ganda Amazon S3.

Untuk informasi lebih lanjut tentang permintaan menggunakan REST API, lihat topik di bawah ini.

Topik

- Bucket dengan hosting virtual
- Meminta pengalihan dan API REST

Bucket dengan hosting virtual

Hosting virtual adalah praktik melayani beberapa situs web dari server web tunggal. Salah satu cara untuk membedakan situs dalam permintaan API Amazon S3 REST Anda adalah dengan menggunakan nama host yang jelas dari Request-URI, bukan hanya bagian nama jalur dari URI. Permintaan biasa Amazon S3 REST menentukan bucket dengan menggunakan komponen slash-delimited pertama dari jalur URI-Permintaan. Sebagai gantinya, Anda dapat menggunakan hosting virtual Amazon S3 untuk menangani bucket dalam panggilan REST API dengan menggunakan header HTTP`Host`. Dalam praktiknya, Amazon S3 menafsirkan `Host` sebagai arti bahwa sebagian besar bucket secara otomatis dapat diakses untuk jenis permintaan terbatas di `https://`*`bucket-name`*`.s3.`*`region-code`*`.amazonaws.com`. Untuk daftar lengkap Wilayah dan titik akhir Amazon S3, lihat titik akhir dan kuota Amazon S3 di. Referensi Umum Amazon Web Services

Hosting virtual juga memiliki manfaat lain. Dengan menamai bucket Anda setelah nama domain terdaftar Anda dan dengan membuat nama DNS alias untuk Amazon S3, Anda dapat sepenuhnya menyesuaikan URL sumber daya Amazon S3, sebagai contoh, `http://my.`*`bucket-name`*`.com/`. Anda juga dapat menerbitkan ke “direktori root” dari server virtual bucket Anda. Kemampuan ini dapat menjadi penting karena banyak aplikasi yang ada mencari berkas di lokasi standar ini. Misalnya,`favicon.ico`,`robots.txt`, dan `crossdomain.xml` semuanya diharapkan dapat ditemukan di root.

> **Important**
>
> Saat Anda menggunakan bucket bergaya host virtual dengan SSL, sertifikat wildcard SSL hanya cocok dengan bucket yang tidak berisi dots (). `.` Untuk mengatasi batasan ini, gunakan

HTTP atau tulis logika verifikasi sertifikat Anda sendiri. Untuk informasi selengkapnya, lihat Rencana Penghentian Jalur Amazon S3 di Blog Berita. AWS

Topik

- Permintaan gaya jalur
- Permintaan bergaya host virtual
- Spesifikasi bucket Host header HTTP
- Contoh
- Menyesuaikan URL Amazon S3 dengan catatan CNAME
- Cara mengaitkan nama host dengan bucket Amazon S3
- Keterbatasan:
- Kompatibilitas mundur

Permintaan gaya jalur

Saat ini, Amazon S3 mendukung akses URL gaya host virtual dan gaya jalur secara keseluruhan. Wilayah AWS Namun, URL gaya jalur akan dihentikan di masa depan. Untuk informasi selengkapnya, lihat Catatan penting berikut ini.

Di Amazon S3, URL gaya jalur menggunakan format berikut:

```
https://s3.region-code.amazonaws.com/bucket-name/key-name
```

Misalnya, jika Anda membuat bucket dengan nama `DOC-EXAMPLE-BUCKET1` di Wilayah Barat AS (Oregon), dan Anda ingin mengakses objek `puppy.jpg` dalam bucket tersebut, Anda dapat menggunakan URL cara jalur berikut:

```
https://s3.us-west-2.amazonaws.com/DOC-EXAMPLE-BUCKET1/puppy.jpg
```

> **Important**
>
> Pembaruan (23 September 2020) — Untuk memastikan bahwa pelanggan memiliki waktu yang mereka butuhkan untuk beralih ke URL bergaya host virtual, kami telah memutuskan

untuk menunda penghentian URL gaya jalur. Untuk informasi selengkapnya, lihat Rencana Penghentian Jalur Amazon S3 – Kisah Selanjutnya dalam Blog Berita AWS.

> **Warning**
>
> Saat menghosting konten situs web yang akan diakses dari browser web, hindari menggunakan URL gaya jalur, yang mungkin mengganggu model keamanan asal browser yang sama. Untuk meng-host konten situs web, kami menyarankan Anda menggunakan titik akhir situs web S3 atau distribusi. CloudFront Untuk informasi selengkapnya, lihat Titik akhir situs web dan Menerapkan aplikasi satu halaman berbasis Reaksi ke Amazon S3 dan di Pola Panduan Perspektif. CloudFront AWS

Permintaan bergaya host virtual

Dalam URI cara hosting virtual, nama bucket adalah bagian dari nama domain di URL.

URL bergaya host virtual Amazon S3 menggunakan format berikut:

```
https://bucket-name.s3.region-code.amazonaws.com/key-name
```

Dalam contoh ini, `DOC-EXAMPLE-BUCKET1` adalah nama bucket, US West (Oregon) adalah Wilayah, dan `puppy.png` adalah nama kuncinya:

```
https://DOC-EXAMPLE-BUCKET1.s3.us-west-2.amazonaws.com/puppy.png
```

Spesifikasi bucket **`Host`** header HTTP

Selama permintaan `GET` Anda tidak menggunakan titik akhir SSL, Anda dapat menentukan bucket untuk permintaan menggunakan header HTTP `Host`. Header `Host` dalam permintaan REST diinterpretasikan sebagai berikut:

- Jika header `Host` dihilangkan atau nilainya `s3.`*`region-code`*`.amazonaws.com`, bucket untuk permintaan akan menjadi komponen slash-delimited pertama dari URI-Permintaan, dan kunci untuk permintaan tersebut adalah sisa dari URI-Permintaan. Ini adalah metode biasa, seperti yang digambarkan oleh contoh pertama dan kedua di bagian ini. Menghilangkan `Host` header hanya berlaku untuk permintaan HTTP 1.0.

- Atau, jika nilai header `Host` berakhir dalam `.s3.`*`region-code`*`.amazonaws.com`, nama bucket adalah komponen utama nilai header `Host` hingga `.s3.`*`region-code`*`.amazonaws.com`. Kunci untuk permintaan adalah Permintaan-URI. Interpretasi ini memaparkan bucket sebagai subdomain `.s3.`*`region-code`*`.amazonaws.com`, sebagaimana digambarkan oleh contoh ketiga dan keempat dalam bagian ini.
- Jika tidak, bucket permintaan adalah nilai huruf kecil dari header `Host`, dan kunci untuk permintaan adalah URI-Permintaan. Interpretasi ini berguna ketika Anda telah mendaftarkan nama DNS yang sama dengan nama bucket Anda dan telah mengonfigurasi nama itu menjadi nama kanonik (CNAME) alias untuk Amazon S3. Prosedur untuk mendaftarkan nama domain dan mengonfigurasi catatan DNS CNAME berada di luar cakupan panduan ini, tetapi hasilnya diilustrasikan oleh contoh terakhir di bagian ini.

Contoh

Bagian ini memberikan URL contoh dan permintaan.

Example — URL dan permintaan gaya jalur

Contoh ini menggunakan hal berikut:

- Nama Bucket - `example.com`
- Wilayah - AS Timur (N. Virginia)
- Nama Kunci - `homepage.html`

URL-nya adalah sebagai berikut:

```
http://s3.us-east-1.amazonaws.com/example.com/homepage.html
```

Permintaan tersebut adalah sebagai berikut:

```
GET /example.com/homepage.html HTTP/1.1
Host: s3.us-east-1.amazonaws.com
```

Permintaan dengan HTTP 1.0 dan menghilangkan header `Host` adalah sebagai berikut:

```
GET /example.com/homepage.html HTTP/1.0
```

Untuk informasi tentang nama yang kompatibel dengan DNS, lihat Batasan. Untuk informasi lebih lanjut tentang kunci, lihat Kunci.

Example — URL dan permintaan bergaya host virtual

Contoh ini menggunakan hal berikut:

- Nama ember - `DOC-EXAMPLE-BUCKET1`
- Wilayah - Eropa (Irlandia)
- Nama kunci - `homepage.html`

URL-nya adalah sebagai berikut:

```
http://DOC-EXAMPLE-BUCKET1.s3.eu-west-1.amazonaws.com/homepage.html
```

Permintaan tersebut adalah sebagai berikut:

```
GET /homepage.html HTTP/1.1
Host: DOC-EXAMPLE-BUCKET1.s3.eu-west-1.amazonaws.com
```

Example — Metode alias CNAME

Untuk menggunakan metode ini, Anda harus mengonfigurasi nama DNS Anda sebagai CNAME alias untuk *`bucket-name`*`.s3.us-east-1.amazonaws.com`. Untuk informasi selengkapnya, lihat Menyesuaikan URL Amazon S3 dengan catatan CNAME.

Contoh ini menggunakan hal berikut:

- Nama Bucket - `example.com`
- Nama kunci - `homepage.html`

URL-nya adalah sebagai berikut:

```
http://www.example.com/homepage.html
```

Contohnya adalah sebagai berikut:

```
GET /homepage.html HTTP/1.1
```

```
Host: www.example.com
```

Menyesuaikan URL Amazon S3 dengan catatan CNAME

Bergantung pada kebutuhan Anda, Anda mungkin tidak ingin `s3.region-code.amazonaws.com` agar muncul di situs web atau layanan Anda. Sebagai contoh, jika Anda meng-hosting gambar situs web di Amazon S3, Anda mungkin lebih memilih `http://images.example.com/` alih-alih `http://images.example.com.s3.us-east-1.amazonaws.com/`. Setiap bucket dengan nama yang kompatibel dengan DNS dapat direferensikan sebagai berikut: `http://BucketName.s3.Region.amazonaws.com/[Filename]`, sebagai contoh, `http://images.example.com.s3.us-east-1.amazonaws.com/mydog.jpg`. Dengan menggunakan CNAME, Anda dapat memetakan `images.example.com` ke nama host Amazon S3 agar URL sebelumnya dapat menjadi `http://images.example.com/mydog.jpg`.

Nama bucket Anda harus sama dengan CNAME. Misalnya, jika Anda membuat CNAME untuk memetakan `images.example.com` untuk `images.example.com.s3.us-east-1.amazonaws.com`, keduanya `http://images.example.com/filename` dan `http://images.example.com.s3.us-east-1.amazonaws.com/filename` akan tetap sama.

Catatan DNS CNAME harus menghubungkan nama domain Anda dengan nama host cara hosting virtual yang sesuai. Misalnya, jika nama bucket dan nama domain Anda adalah `images.example.com` dan bucket Anda berada di Wilayah Timur AS (N. Virginia), catatan CNAME harus alias ke `images.example.com.s3.us-east-1.amazonaws.com`.

```
images.example.com CNAME    images.example.com.s3.us-east-1.amazonaws.com.
```

Amazon S3 menggunakan nama host untuk menentukan nama bucket. Jadi CNAME dan nama bucket harus sama. Sebagai contoh, anggaplah Anda telah mengonfigurasi `www.example.com` sebagai CNAME untuk `www.example.com.s3.us-east-1.amazonaws.com`. Saat Anda mengakses `http://www.example.com`, Amazon S3 menerima permintaan yang serupa dengan yang berikut ini:

Example

```
GET / HTTP/1.1
Host: www.example.com
Date: date
Authorization: signatureValue
```

Amazon S3 hanya melihat nama host asli `www.example.com` dan tidak menyadari pemetaan CNAME yang digunakan untuk menyelesaikan permintaan.

Anda dapat menggunakan endpoint Amazon S3 apa pun dalam alias CNAME. Misalnya, `s3.ap-southeast-1.amazonaws.com` dapat digunakan dalam alias CNAME. Untuk informasi lebih lanjut tentang titik akhir, lihat Titik akhir Permintaan. Untuk membuat situs web statis dengan menggunakan domain khusus, lihatTutorial: Mengonfigurasi situs web statis menggunakan domain kustom yang terdaftar di Route 53.

> **Important**
>
> Saat menggunakan URL khusus dengan CNames, Anda harus memastikan ada bucket yang cocok untuk catatan CNAME atau alias apa pun yang Anda konfigurasikan. Misalnya, jika Anda membuat entri DNS untuk `www.example.com` dan `login.example.com` mempublikasikan konten web menggunakan S3, Anda harus membuat bucket dan. `www.example.com` `login.example.com`
> Saat catatan CNAME atau alias dikonfigurasi dengan menunjuk ke titik akhir S3 tanpa bucket yang cocok, AWS pengguna mana pun dapat membuat bucket tersebut dan memublikasikan konten di bawah alias yang dikonfigurasi, meskipun kepemilikannya tidak sama.
> Untuk alasan yang sama, sebaiknya Anda mengubah atau menghapus CNAME atau alias yang sesuai saat menghapus bucket.

Cara mengaitkan nama host dengan bucket Amazon S3

Untuk mengaitkan nama host dengan bucket Amazon S3 dengan menggunakan alias CNAME

1. Pilih nama host milik domain yang Anda kontrol.

   Contoh ini menggunakan subdomain `images` dari domain `example.com`.
2. Buat bucket yang sesuai dengan nama host.

   Dalam contoh ini, nama host dan bucket adalah `images.example.com`. Nama bucket harus sama persis dengan hostname.
3. Buat data DNS CNAME yang mendefinisikan nama host sebagai alias untuk bucket Amazon S3.

   Misalnya:

   ```
   images.example.com CNAME images.example.com.s3.us-west-2.amazonaws.com
   ```

> **Important**
>
> Untuk alasan perutean permintaan, catatan DNS CNAME harus didefinisikan persis seperti yang ditunjukkan pada contoh sebelumnya. Jika tidak, mungkin tampak beroperasi dengan benar, tetapi pada akhirnya akan menghasilkan perilaku yang tidak terduga.

Prosedur untuk mengkonfigurasi catatan DNS CNAME tergantung pada server DNS atau penyedia DNS Anda. Untuk informasi khusus, lihat dokumentasi server Anda atau hubungi penyedia Anda.

Keterbatasan:

Dukungan SOAP melalui HTTP tidak digunakan lagi, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami sarankan Anda menggunakan REST API atau AWS SDK.

Kompatibilitas mundur

Bagian berikut mencakup berbagai aspek kompatibilitas mundur Amazon S3 yang terkait dengan permintaan URL gaya jalur dan gaya host virtual.

Titik akhir warisan

Beberapa Wilayah mendukung titik akhir legasi. Anda mungkin melihat titik akhir ini di log akses server Anda atau log AWS CloudTrail. Untuk informasi lebih lanjut, tinjau informasi berikut. Untuk daftar lengkap Wilayah dan titik akhir Amazon S3, lihat titik akhir dan kuota Amazon S3 di. Referensi Umum Amazon Web Services

> **Important**
>
> Meskipun Anda mungkin melihat titik akhir legasi di log Anda, kami menyarankan Anda untuk selalu menggunakan sintaks titik akhir standar untuk mengakses bucket Anda.
> URL bergaya host virtual Amazon S3 menggunakan format berikut:
>
> ```
> https://bucket-name.s3.region-code.amazonaws.com/key-name
> ```

Di Amazon S3, URL gaya jalur menggunakan format berikut:

```
https://s3.region-code.amazonaws.com/bucket-name/key-name
```

s3-Wilayah

Beberapa Wilayah Amazon S3 yang lebih lama mendukung titik akhir yang berisi tanda hubung (`-`) antara `s3` dan kode Wilayah (misalnya,`s3-us-west-2`), bukan titik (misalnya,). `s3.us-west-2` Jika bucket Anda berada di salah satu Wilayah ini, Anda mungkin melihat format titik akhir berikut dalam log akses server Anda atau log CloudTrail:

```
https://bucket-name.s3-region-code.amazonaws.com
```

Dalam contoh ini, nama bucket adalah DOC-EXAMPLE-BUCKET1 dan Wilayahnya adalah US West (Oregon):

```
https://DOC-EXAMPLE-BUCKET1.s3-us-west-2.amazonaws.com
```

Titik akhir global warisan

Untuk beberapa Wilayah, Anda dapat menggunakan titik akhir global lama untuk membuat permintaan yang tidak menentukan titik akhir khusus Wilayah. Titik titik-akhir global legasi adalah sebagai berikut:

```
bucket-name.s3.amazonaws.com
```

Di log atau CloudTrail log akses server, Anda mungkin melihat permintaan yang menggunakan titik akhir global lama. Dalam contoh ini, nama bucket adalah `DOC-EXAMPLE-BUCKET1` dan titik akhir global lama adalah:

```
https://DOC-EXAMPLE-BUCKET1.s3.amazonaws.com
```

Permintaan bergaya host virtual untuk AS Timur (Virginia N.)

Permintaan yang dibuat dengan titik akhir global lama masuk ke Wilayah AS Timur (Virginia N.) secara default. Oleh karena itu, titik akhir global legasi terkadang digunakan sebagai pengganti titik akhir Wilayah untuk Timur AS (N. Virginia). Jika Anda membuat bucket di Timur AS (N. Virginia) dan

menggunakan titik akhir global, Amazon S3 mengirimkan permintaan Anda ke Wilayah ini secara default.

Permintaan bergaya host virtual untuk Wilayah lain

Titik akhir global lama juga digunakan untuk permintaan gaya host virtual di Wilayah lain yang didukung. Jika Anda membuat bucket di Wilayah yang diluncurkan sebelum 20 Maret 2019, dan menggunakan titik akhir global lama, Amazon S3 memperbarui catatan DNS untuk mengalihkan permintaan ke lokasi yang benar, yang mungkin membutuhkan waktu. Sementara itu, aturan default berlaku, dan permintaan gaya host virtual Anda masuk ke Wilayah AS Timur (Virginia N.). Amazon S3 kemudian mengalihkannya dengan Pengalihan Sementara HTTP 307 ke Wilayah yang benar.

Untuk bucket S3 di Wilayah yang diluncurkan setelah 20 Maret 2019, server DNS tidak merutekan permintaan Anda langsung ke Wilayah AWS tempat bucket Anda berada. Itu mengembalikan pesan kesalahan HTTP 400 Bad Request. Untuk informasi selengkapnya, lihat Membuat permintaan.

Permintaan gaya jalur

Untuk Wilayah AS Timur (Virginia N.), Anda dapat menggunakan titik akhir global lama untuk permintaan gaya jalur.

Untuk semua Wilayah lainnya, sintaks cara jalur mengharuskan Anda menggunakan titik akhir spesifik-Wilayah saat mencoba mengakses bucket. Jika Anda mencoba mengakses bucket dengan titik akhir global lama atau titik akhir lain yang berbeda dari yang untuk Wilayah tempat bucket berada, Anda akan menerima kesalahan Pengalihan Sementara kode respons HTTP 307 dan pesan yang menunjukkan URI yang benar untuk sumber daya Anda. Misalnya, jika Anda menggunakan `https://s3.amazonaws.com/bucket-name` untuk bucket yang dibuat di Wilayah Barat AS (Oregon), Anda akan menerima pesan kesalahan HTTP 307 Temporary Redirect.

Meminta pengalihan dan API REST

Topik

- Pengalihan dan agen pengguna HTTP
- Mengarahkan kembali dan 100-Lanjutkan
- Contoh pengalihan

Bagian ini menjelaskan cara menangani pengalihan HTTP menggunakan API REST Amazon S3. Untuk informasi umum tentang pengalihan Amazon S3, lihat Membuat permintaan dalam Referensi API Amazon Simple Storage Service.

Pengalihan dan agen pengguna HTTP

Program yang menggunakan API REST Amazon S3 harus menangani pengalihan baik pada lapisan aplikasi atau lapisan HTTP. Banyak pustaka klien dan agen pengguna HTTP dapat dikonfigurasi agar menangani pengalihan secara otomatis dengan tepat; tetapi, banyak klien lain yang memiliki implementasi pengalihan yang tidak tepat atau tidak lengkap.

Sebelum Anda mengandalkan pustaka untuk memenuhi persyaratan pengalihan, uji kasus berikut:

- Verifikasi bahwa semua header permintaan HTTP disertakan dengan benar dalam permintaan pengalihan (permintaan kedua setelah menerima pengalihan) termasuk standar HTTP seperti Otorisasi dan Tanggal.
- Verifikasi pengalihan non-GET, seperti PUT dan DELETE, bekerja dengan benar.
- Verifikasi permintaan PUT besar mengikuti pengalihan dengan benar.
- Verifikasi permintaan PUT mengikuti arah yang benar jika respons 100-lanjutkan memerlukan waktu lama untuk tiba.

Agen pengguna HTTP yang secara ketat mematuhi RFC 2616 mungkin memerlukan konfirmasi eksplisit sebelum mengikuti pengalihan jika metode permintaan HTTP bukan GET atau HEAD. Pada umumnya, aman untuk mengikuti pengalihan yang dihasilkan oleh Amazon S3 secara otomatis, karena sistem akan menerbitkan pengalihan hanya untuk host di domain amazonaws.com dan efek permintaan pengalihan akan sama dengan permintaan awal.

Mengarahkan kembali dan 100-Lanjutkan

Untuk menyederhanakan penanganan pengalihan, meningkatkan efisiensi, dan menghindari biaya yang terkait dengan pengiriman bodi permintaan pengalihan dua kali, konfigurasikan aplikasi Anda untuk menggunakan 100-lanjutkan untuk operasi PUT. Ketika aplikasi Anda menggunakan 100-lanjutkan, aplikasi tidak mengirim bodi permintaan sampai menerima pengakuan. Jika pesan ditolak berdasarkan header, bodi pesan tidak akan terkirim. Untuk informasi lebih lanjut tentang 100-lanjutkan, kunjungi RFC 2616 Bagian 8.2.3

> **Note**
>
> Menurut RFC 2616, saat menggunakan `Expect: Continue` dengan server HTTP yang tidak dikenal, Anda tidak boleh menunggu selama periode yang tidak terbatas sebelum mengirimkan bodi permintaan. Hal ini karena beberapa server HTTP tidak mengenali 100-lanjutkan. Namun, Amazon S3 mengakui jika permintaan Anda memuat `Expect:`

`Continue` dan akan merespons dengan status 100-lanjutkan sementara atau kode status akhir. Selain itu, tidak ada kesalahan pengalihan yang akan terjadi setelah menerima go-ahead sementara 100 lanjutkan. Ini akan membantu Anda menghindari menerima respons pengalihan saat Anda masih menulis bodi permintaan.

Contoh pengalihan

Bagian ini memberikan contoh interaksi server-klien menggunakan pengalihan HTTP dan 100-lanjutkan.

Berikut ini adalah contoh PUT ke bucket `quotes.s3.amazonaws.com`.

```
PUT /nelson.txt HTTP/1.1
Host: quotes.s3.amazonaws.com
Date: Mon, 15 Oct 2007 22:18:46 +0000

Content-Length: 6
Expect: 100-continue
```

Amazon S3 mengembalikan hal berikut:

```
HTTP/1.1 307 Temporary Redirect
Location: http://quotes.s3-4c25d83b.amazonaws.com/nelson.txt?rk=8d47490b
Content-Type: application/xml
Transfer-Encoding: chunked
Date: Mon, 15 Oct 2007 22:18:46 GMT

Server: AmazonS3

<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>TemporaryRedirect</Code>
  <Message>Please re-send this request to the
  specified temporary endpoint. Continue to use the
  original request endpoint for future requests.
  </Message>
  <Endpoint>quotes.s3-4c25d83b.amazonaws.com</Endpoint>
  <Bucket>quotes</Bucket>
</Error>
```

Klien mengikuti respons pengalihan dan mengeluarkan permintaan baru ke titik akhir sementara `quotes.s3-4c25d83b.amazonaws.com`.

```
PUT /nelson.txt?rk=8d47490b HTTP/1.1
Host: quotes.s3-4c25d83b.amazonaws.com
Date: Mon, 15 Oct 2007 22:18:46 +0000

Content-Length: 6
Expect: 100-continue
```

Amazon S3 mengembalikan 100-lanjutkan yang menunjukkan bahwa klien harus melanjutkan dengan mengirim bodi permintaan.

```
HTTP/1.1 100 Continue
```

Klien mengirimkan bodi permintaan.

```
ha ha\n
```

Amazon S3 mengembalikan respons akhir.

```
HTTP/1.1 200 OK
Date: Mon, 15 Oct 2007 22:18:48 GMT

ETag: "a2c8d6b872054293afd41061e93bc289"
Content-Length: 0
Server: AmazonS3
```

Mengembangkan dengan Amazon S3 menggunakan AWS CLI

Ikuti langkah-langkah berikut untuk mengunduh dan mengonfigurasi AWS Command Line Interface (AWS CLI).

Untuk daftar perintah AWS CLI Amazon S3, lihat halaman berikut di Referensi PerintahAWS CLI:

- s3
- s3api
- s3control

> **Note**
>
> Layanan di AWS, seperti Amazon S3, mewajibkan Anda untuk memberikan kredensial saat mengaksesnya. Layanan tersebut kemudian menentukan apakah Anda memiliki izin untuk mengakses sumber daya yang dimiliki oleh layanan. Konsol tersebut memerlukan kata sandi. Anda dapat membuat kunci akses untuk Akun AWS Anda agar dapat mengakses AWS CLI atau API. Namun, kami tidak menyarankan Anda untuk mengakses AWS dengan menggunakan kredensial untuk Akun AWS. Sebagai gantinya, sebaiknya gunakan (IAM) AWS Identity and Access Management. Buat pengguna IAM, tambahkan pengguna tersebut ke grup IAM dengan izin administratif, lalu berikan izin administratif ke pengguna IAM yang sudah dibuat. Kemudian, Anda dapat mengakses AWS dengan menggunakan URL khusus dan kredensial pengguna IAM tersebut. Untuk instruksinya, buka Membuat Grup Administrator dan pengguna IAM Pertama Anda di Panduan Pengguna IAM.

Untuk menyiapkan AWS CLI

1. Unduh dan konfigurasikan AWS CLI. Untuk instruksinya, lihat topik berikut di AWS Command Line InterfacePanduan Pengguna:

   - Mempersiapkan AWS Command Line Interface
   - Mengonfigurasi AWS Command Line Interface

2. Tambahkan profil bernama untuk pengguna administrator di dalam file konfig AWS CLI. Anda dapat menggunakan profil ini saat menjalankan perintah AWS CLI. Untuk informasi selengkapnya, lihat Profil bernama untuk AWS CLI di dalam AWS Command Line InterfacePanduan Pengguna.

   ```
   [adminuser]
   aws_access_key_id = adminuser access key ID
   aws_secret_access_key = adminuser secret access key
   region = aws-region
   ```

   Untuk daftar yang tersediaWilayah AWS, lihat Wilayah dan Titik Akhir di Referensi Umum AWS.

3. Verifikasikan pengaturan dengan mengetikkan perintah berikut di command prompt.

   - Coba perintah `help` untuk memverifikasi bahwa AWS CLI sudah terpasang di komputer Anda:

```
aws help
```

- Jalankan perintah `S3` dengan menggunakan kredensial `adminuser` yang baru saja dibuat. Untuk melakukannya, tambahkan parameter `--profile` ke perintah Anda untuk menentukan nama profil. Di contoh ini, perintah `ls` mencantumkan bucket di akun Anda. AWS CLI menggunakan kredensial `adminuser` untuk mengautentikasi permintaan.

```
aws s3 ls --profile adminuser
```

Mengembangkan dengan Amazon S3 menggunakan SDK AWS

AWS kit pengembangan perangkat lunak (SDK) tersedia untuk banyak bahasa pemrograman populer. Setiap SDK menyediakan API, contoh kode, dan dokumentasi yang memudahkan developer untuk membangun aplikasi dalam bahasa pilihan mereka.

> **Note**
>
> Anda dapat menggunakan AWS Amplify untuk pengembangan end-to-end fullstack aplikasi web dan seluler. Amplify Storage mengintegrasikan kemampuan penyimpanan dan manajemen file dengan mulus ke web frontend dan aplikasi seluler, yang dibangun di atas Amazon S3. Untuk informasi selengkapnya, lihat Penyimpanan di panduan pengguna Amplify.

Menggunakan layanan ini dengan AWS SDK

AWS kit pengembangan perangkat lunak (SDK) tersedia untuk banyak bahasa pemrograman populer. Setiap SDK menyediakan API, contoh kode, dan dokumentasi yang memudahkan developer untuk membangun aplikasi dalam bahasa pilihan mereka.

Dokumentasi SDK	Contoh kode
AWS SDK for C++	AWS SDK for C++ contoh kode
AWS CLI	AWS CLI contoh kode

Dokumentasi SDK	Contoh kode
AWS SDK for Go	AWS SDK for Go contoh kode
AWS SDK for Java	AWS SDK for Java contoh kode
AWS SDK for JavaScript	AWS SDK for JavaScript contoh kode
AWS SDK for Kotlin	AWS SDK for Kotlin contoh kode
AWS SDK for .NET	AWS SDK for .NET contoh kode
AWS SDK for PHP	AWS SDK for PHP contoh kode
AWS Tools for PowerShell	Alat untuk contoh PowerShell kode
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) contoh kode
AWS SDK for Ruby	AWS SDK for Ruby contoh kode
AWS SDK for Rust	AWS SDK for Rust contoh kode
AWS SDK untuk SAP ABAP	AWS SDK untuk SAP ABAP contoh kode
AWS SDK for Swift	AWS SDK for Swift contoh kode

Untuk contoh khusus untuk layanan ini, lihat Contoh kode untuk Amazon S3 menggunakan SDK AWS.

> **Ketersediaan contoh**
>
> Tidak dapat menemukan apa yang Anda butuhkan? Minta contoh kode menggunakan tautan Berikan umpan balik pada bagian bawah halaman ini.

Antarmuka Pemrograman SDK

Setiap AWS SDK menyediakan satu atau lebih antarmuka terprogram untuk bekerja dengan Amazon S3. Setiap SDK menyediakan antarmuka tingkat rendah untuk Amazon S3, dengan metode yang sangat mirip dengan operasi API. Beberapa SDK menyediakan antarmuka tingkat tinggi

untuk Amazon S3, yang merupakan abstraksi yang dimaksudkan untuk menyederhanakan kasus penggunaan umum.

Misalnya, saat Anda melakukan pengunggahan multibagian menggunakan operasi API tingkat rendah, Anda perlu menggunakan operasi untuk memulai unggahan, operasi lain untuk mengunggah komponen, dan operasi akhir untuk menyelesaikan unggahan. Operasi API unggahan multibagian tingkat tinggi memungkinkan Anda melakukan semua operasi yang diperlukan untuk mengunggah dalam satu panggilan API. Sebagai contoh, lihat Pengunggahan objek menggunakan unggahan multibagian.

Operasi API tingkat rendah memungkinkan kontrol yang lebih besar atas unggahan. Kami menyarankan Anda menggunakan operasi API tingkat rendah jika Anda perlu menjeda dan melanjutkan unggahan, memvariasikan ukuran bagian selama pengunggahan, atau mulai mengunggah saat Anda tidak mengetahui ukuran data sebelumnya.

Menentukan Versi Tanda Tangan di Autentikasi Permintaan

Amazon S3 hanya mendukung AWS Signature Version 4 di sebagian besar. Wilayah AWS Di beberapa yang lebih lama Wilayah AWS, Amazon S3 mendukung Signature Version 4 dan Signature Version 2. Akan tetapi, Signature Version 2 sedang dinonaktifkan (dihentikan). Untuk informasi lebih lanjut tentang akhir dukungan untuk Signature Version 2, lihat AWS Tanda Tangan Versi 2 Dimatikan (Tidak Digunakan Lagi) untuk Amazon S3.

Untuk daftar dari semua Wilayah Amazon S3 dan versi tanda tangan yang didukung, lihat Wilayah dan Titik Akhir dalam Referensi Umum AWS .

Untuk semua Wilayah AWS, AWS SDK menggunakan Signature Version 4 secara default untuk mengautentikasi permintaan. Saat menggunakan AWS SDK yang dirilis sebelum Mei 2016, Anda mungkin diminta untuk meminta Tanda Tangan Versi 4, seperti yang ditunjukkan pada tabel berikut.

SDK	Meminta Signature Version 4 untuk Autentikasi Permintaan
AWS CLI	Untuk profil default, jalankan perintah berikut: `$ aws configure set default.s3.signature_version s3v4` Untuk profil kustom, jalankan perintah berikut:

SDK	Meminta Signature Version 4 untuk Autentikasi Permintaan
	`$ aws configure set profile.your_profile_name.s3.signature_version s3v4`

SDK Java	Tambahkan hal berikut di kode Anda:
	`System.setProperty(SDKGlobalConfiguration.ENABLE_S3_SIGV4_SYSTEM_PROPERTY, "true");`
	Atau, pada baris perintah, tentukan hal berikut:
	`-Dcom.amazonaws.services.s3.enableV4`

JavaScript SDK	Tetapkan parameter `signatureVersion` ke `v4` saat mengonstruksi klien: `var s3 = new AWS.S3({signatureVersion: 'v4'});`

SDK PHP	Tetapkan parameter `signature` ke `v4` saat mengonstruksi klien layanan Amazon S3 untuk PHP SDK v2:
	`<?php`
	`$client = S3Client::factory([`
	`  'region' => 'YOUR-REGION',`
	`  'version' => 'latest',`
	`  'signature' => 'v4'`
	`]);`
	Saat menggunakan PHP SDK v3, atur parameter `signature_version` ke `v4` selama konstruksi klien layanan Amazon S3:
	`<?php`
	`$s3 = new Aws\S3\S3Client([`
	`  'version' => '2006-03-01',`
	`  'region' => 'YOUR-REGION',`
	`  'signature_version' => 'v4'`
	`]);`

SDK	Meminta Signature Version 4 untuk Autentikasi Permintaan
SDK Python-Boto	Tentukan hal berikut dalam berkas konfigurasi default boto: `[s3] use-sigv4 = True`

Ruby SDK	Ruby SDK-Versi 1: Atur parameter `:s3_signature_version` ke `:v4` saat mengonstruksi klien:
	`s3 = AWS::S3::Client.new(:s3_signature_version => :v4)`
	Ruby SDK-Versi 3: Atur parameter `signature_version` ke `v4` saat mengonstruksi klien:
	`s3 = Aws::S3::Client.new(signature_version: 'v4')`

SDK .NET	Tambahkan yang berikut ini ke kode sebelum membuat klien Amazon S3:
	`AWSConfigsS3.UseSignatureVersion4 = true;`
	Atau, tambahkan yang berikut ini ke file konfigurasi:
	`<appSettings> <add key="AWS.S3.UseSignatureVersion4" value="true" /> </appSettings>`

AWS Tanda Tangan Versi 2 Dimatikan (Tidak Digunakan Lagi) untuk Amazon S3

Signature Version 2 dinonaktifkan (dihentikan) di Amazon S3. Amazon S3 kemudian hanya akan menerima permintaan API yang ditandatangani menggunakan Signature Version 4.

Bagian ini memberikan jawaban atas pertanyaan umum mengenai akhir dukungan untuk Signature Version 2.

Apa yang dimaksud dengan Signature Version 2/4, dan Apa Artinya Menandatangani Permintaan?

Proses penandatanganan Signature Version 2 atau Signature Version 4 digunakan untuk mengautentikasi permintaan Amazon S3 API Anda. Permintaan penandatanganan memungkinkan Amazon S3 mengidentifikasi siapa yang mengirim permintaan dan melindungi permintaan Anda dari pelaku kejahatan.

Untuk informasi selengkapnya tentang AWS permintaan penandatanganan, lihat Menandatangani Permintaan AWS API di Referensi Umum AWS.

Pembaruan Apa yang Anda Buat?

Saat ini kami mendukung permintaan API Amazon S3 yang ditandatangani menggunakan proses Signature Version 2 dan Signature Version 4. Setelah itu, Amazon S3 kemudian hanya akan menerima permintaan yang ditandatangani menggunakan Signature Version 4.

Untuk informasi selengkapnya tentang AWS permintaan penandatanganan, lihat Perubahan dalam Versi Tanda Tangan 4 di Referensi Umum AWS.

Mengapa Anda Membuat Pembaruan?

Signature Version 4 memberikan keamanan yang lebih baik dengan menggunakan kunci penandatanganan dan bukan kunci akses rahasia Anda. Signature Version 4 saat ini didukung di semua Wilayah AWS, sedangkan Signature Version 2 hanya didukung di Wilayah yang diluncurkan sebelum Januari 2014. Pembaruan ini memungkinkan kami memberikan pengalaman yang lebih konsisten di seluruh Wilayah.

Bagaimana Saya Memastikan bahwa Saya Menggunakan Signature Version 4, dan Pembaruan Apa yang Saya Perlukan?

Versi tanda tangan yang digunakan untuk menandatangani permintaan Anda biasanya ditetapkan oleh alat atau SDK di sisi klien. Secara default, versi terbaru AWS SDK kami menggunakan Signature Version 4. Untuk perangkat lunak pihak ketiga, hubungi tim dukungan yang sesuai untuk perangkat lunak Anda guna mengonfirmasi versi apa yang Anda butuhkan. Jika Anda mengirim panggilan REST langsung ke Amazon S3, Anda harus memodifikasi aplikasi Anda untuk menggunakan proses penandatanganan Signature Version 4.

Untuk informasi tentang versi AWS SDK yang akan digunakan saat pindah ke Signature Version 4, lihatBeralih dari Signature Version 2 ke Signature Version 4.

Untuk informasi tentang menggunakan Signature Version 4 dengan API REST Amazon S3, lihat Permintaan Autentikasi (Signature Version 4 AWS) di Referensi API Amazon Simple Storage Service.

Apa yang Terjadi jika Saya Tidak Melakukan Pembaruan?

Permintaan yang ditandatangani dengan Signature Version 2 yang dibuat setelahnya akan gagal diautentikasi dengan Amazon S3. Pemohon akan melihat kesalahan yang menyatakan bahwa permintaan harus ditandatangani dengan Signature Version 4.

Haruskah Saya Melakukan Perubahan Meskipun Saya Menggunakan URL yang Ditandatangani Sebelumnya yang Harus Saya Tandatangani Selama Lebih dari 7 Hari?

Jika Anda menggunakan URL yang ditandatangani sebelumnya yang mengharuskan Anda menandatangani lebih dari 7 hari, saat ini tidak ada tindakan yang diperlukan. Anda dapat terus menggunakan AWS Signature Version 2 untuk menandatangani dan mengautentikasi URL yang telah ditetapkan sebelumnya. Kami akan menindaklanjuti dan memberikan detail lebih lanjut tentang cara memigrasi ke Signature Version 4 untuk skenario URL yang ditandatangani sebelumnya.

Info Selengkapnya

- Untuk informasi selengkapnya tentang penggunaan Tanda Tangan Versi 4, lihat Menandatangani Permintaan AWS API.
- Lihat daftar perubahan antara Signature Version 2 dan Signature Version 4 di Perubahan dalam Signature Version 4.
- Lihat postingan AWS Signature Version 4 untuk menggantikan AWS Signature Version 2 untuk menandatangani permintaan API Amazon S3 di AWS forum.
- Jika Anda memiliki pertanyaan atau kekhawatiran, hubungi AWS Support.

Beralih dari Signature Version 2 ke Signature Version 4

Jika saat ini Anda menggunakan Signature Version 2 untuk autentikasi permintaan API Amazon S3, Anda harus beralih ke menggunakan Signature Version 4. Dukungan berakhir untuk Signature

Version 2, sebagaimana dijelaskan di AWS Tanda Tangan Versi 2 Dimatikan (Tidak Digunakan Lagi) untuk Amazon S3.

Untuk informasi tentang menggunakan Signature Version 4 dengan API REST Amazon S3, lihat Permintaan Autentikasi (Signature Version 4 AWS) di Referensi API Amazon Simple Storage Service.

Tabel berikut mencantumkan SDK dengan versi minimum yang diperlukan untuk menggunakan Signature Version 4 (SigV4). Jika Anda menggunakan URL presigned dengan SDK AWS Java, JavaScript (Node.js), atau Python (Boto/CLI), Anda harus mengatur yang benar Wilayah AWS dan mengatur Signature Version 4 dalam konfigurasi klien. Untuk informasi tentang pengaturan `SigV4` di konfigurasi klien, lihat Menentukan Versi Tanda Tangan di Autentikasi Permintaan.

Jika Anda menggunakan SDK/ Produk ini	Tingkatkan ke versi SDK ini	Perubahan kode yang diperlukan bagi klien untuk menggunakan Sigv4?	Tautan ke dokumentasi SDK
AWS SDK for Java v1	Peningkatan ke Java 1.11.201+ atau v2.	Ya	Menentukan Versi Tanda Tangan di Autentikasi Permintaan
AWS SDK for Java v2	Tidak diperlukan peningkatan SDK.	Tidak	AWS SDK for Java
AWS SDK for .NET v1	Tingkatkan ke 3.1.10 atau lebih baru.	Ya	AWS SDK for .NET
AWS SDK for .NET v2	Tingkatkan ke 3.1.10	Tidak	AWS SDK for .NET v2

Jika Anda menggunakan SDK/ Produk ini	Tingkatkan ke versi SDK ini	Perubahan kode yang diperlukan bagi klien untuk menggunakan Sigv4?	Tautan ke dokumentasi SDK
	atau lebih baru.		
AWS SDK for .NET v3	Tingkatkan ke 3.3.0.0 atau lebih baru.	Ya	AWS SDK for .NET v3
AWS SDK for JavaScript v1	Tingkatkan ke 2.68.0 atau lebih baru.	Ya	AWS SDK for JavaScript
AWS SDK for JavaScript v2	Tingkatkan ke 2.68.0 atau lebih baru.	Ya	AWS SDK for JavaScript
AWS SDK for JavaScript v3	Saat ini tidak ada tindakan yang diperlukan. Tingkatkan ke versi utama V3 di Q3 2019.	Tidak	AWS SDK for JavaScript

Jika Anda menggunakan SDK/ Produk ini	Tingkatkan ke versi SDK ini	Perubahan kode yang diperlukan bagi klien untuk menggunakan Sigv4?	Tautan ke dokumentasi SDK
AWS SDK for PHP v1	Rekomendasikan untuk meningkatkan ke versi terbaru PHP atau, setidaknya ke v2.7.4 dengan parameter tanda tangan yang diatur ke v4 dalam konfigurasi klien S3.	Ya	AWS SDK for PHP

Jika Anda menggunakan SDK/ Produk ini	Tingkatkan ke versi SDK ini	Perubahan kode yang diperlukan bagi klien untuk menggunakan Sigv4?	Tautan ke dokumentasi SDK
AWS SDK for PHP v2	Rekomendasikan untuk meningkatkan ke versi terbaru PHP atau, setidaknya ke v2.7.4 dengan parameter tanda tangan yang diatur ke v4 dalam konfigurasi klien S3.	Tidak	AWS SDK for PHP
AWS SDK for PHP v3	Tidak diperlukan peningkatan SDK.	Tidak	AWS SDK for PHP
Boto2	Tingkatkan ke Boto2 v2.49.0.	Ya	Peningkatan Boto 2
Boto3	Tingkatkan ke 1.5.71 (Botocore), 1.4.6 (Boto3).	Ya	Boto 3 - AWS SDK untuk Python

Jika Anda menggunakan SDK/ Produk ini	Tingkatkan ke versi SDK ini	Perubahan kode yang diperlukan bagi klien untuk menggunakan Sigv4?	Tautan ke dokumentasi SDK
AWS CLI	Memutakhirkan ke 1.11.108.	Ya	AWS Command Line Interface
AWS CLI v2 (pratinjau)	Tidak diperlukan peningkatan SDK.	Tidak	AWS Command Line Interface versi 2
AWS SDK for Ruby v1	Tingkatkan ke Ruby V3.	Ya	Ruby V3 untuk AWS
AWS SDK for Ruby v2	Tingkatkan ke Ruby V3.	Ya	Ruby V3 untuk AWS
AWS SDK for Ruby v3	Tidak diperlukan peningkatan SDK.	Tidak	Ruby V3 untuk AWS
Go	Tidak diperlukan peningkatan SDK.	Tidak	AWS SDK for Go
C++	Tidak diperlukan peningkatan SDK.	Tidak	AWS SDK for C++

AWS Tools for Windows PowerShell atau AWS Tools for PowerShell Core

Jika Anda menggunakan versi modul lebih awal dari 3.3.0.0, Anda harus meningkatkan ke 3.3.0.0.

Untuk mendapatkan informasi versi, gunakan cmdlet `Get-Module`:

```
Get-Module -Name AWSPowershell
Get-Module -Name AWSPowershell.NetCore
```

Untuk memperbarui versi 3.3.0.0, gunakan cmdlet `Update-Module`:

```
Update-Module -Name AWSPowershell
Update-Module -Name AWSPowershell.NetCore
```

Anda dapat menggunakan URL yang telah ditandatangani sebelumnya yang valid selama lebih dari 7 hari yang akan Anda gunakan untuk mengirimkan lalu lintas Signature Version 2.

Berkembang dengan Amazon S3 menggunakan API REST

Arsitektur Amazon S3 dirancang untuk netral bahasa pemrograman, menggunakan antarmuka kami yang didukung untuk menyimpan dan mengambil objek.

Amazon S3 menyediakan antarmuka REST. Dengan REST, metadata dikembalikan dalam header HTTP. Karena kami hanya mendukung permintaan HTTP hingga 4 KB (tidak termasuk bodi), jumlah metadata yang dapat Anda pasok dibatasi. API REST adalah antarmuka HTTP ke Amazon S3. Dengan menggunakan REST, Anda menggunakan permintaan HTTP standar untuk membuat, menarik, dan menghapus bucket dan objek.

Anda dapat menggunakan toolkit yang mendukung HTTP untuk menggunakan API REST. Anda bahkan dapat menggunakan peramban untuk mengambil objek, selama dapat dibaca secara anonim.

API REST menggunakan header HTTP standar dan kode status, sehingga peramban dan toolkit standar bekerja sesuai harapan. Di beberapa area, kami telah menambahkan fungsi ke HTTP (misalnya, kami menambahkan header untuk mendukung kontrol akses). Dalam kasus ini, kami telah

melakukan yang terbaik untuk menambahkan fungsionalitas baru dengan cara yang cocok dengan cara penggunaan HTTP standar.

Untuk informasi selengkapnya tentang mengirim permintaan menggunakan REST API, lihatMembuat permintaan menggunakan API REST. Untuk beberapa pertimbangan yang harus Anda ingat saat menggunakan REST API, lihat topik di bawah ini.

Untuk informasi selengkapnya tentang Amazon S3 REST API, lihat Referensi API Amazon Simple Storage Service.

Topik

- Meminta perutean

Meminta perutean

Program yang membuat permintaan terhadap bucket yang dibuat menggunakan CreateBucketAPI yang mencakup pengalihan CreateBucketConfigurationharus mendukung. Selain itu, beberapa klien yang tidak menghormati DNS TTL dapat menghadapi masalah.

Bagian ini menjelaskan perutean dan masalah DNS untuk dipertimbangkan saat merancang layanan atau aplikasi Anda untuk digunakan dengan Amazon S3.

Minta pengalihan dan API REST

Amazon S3 menggunakan Domain Name System (DNS) untuk merutekan permintaan ke fasilitas yang dapat memprosesnya. Sistem ini bekerja secara efektif, tetapi kesalahan perutean sementara dapat terjadi. Jika permintaan tiba di lokasi Amazon S3 yang salah, Amazon S3 menanggapi dengan pengalihan sementara yang memberi tahu pemohon untuk mengirim ulang permintaan ke titik akhir baru. Jika salah membentuk permintaan, Amazon S3 menggunakan pengalihan permanen untuk memberikan arahan tentang cara menjalankan permintaan dengan benar.

> **Important**
>
> Untuk menggunakan fitur ini, Anda harus memiliki aplikasi yang dapat menangani respons pengalihan Amazon S3. Satu-satunya pengecualian adalah untuk aplikasi yang bekerja secara eksklusif dengan bucket yang dibuat tanpa `<CreateBucketConfiguration>`. Untuk informasi lebih lanjut tentang batasan lokasi, lihat Mengakses dan mendaftarkan bucket Amazon S3.

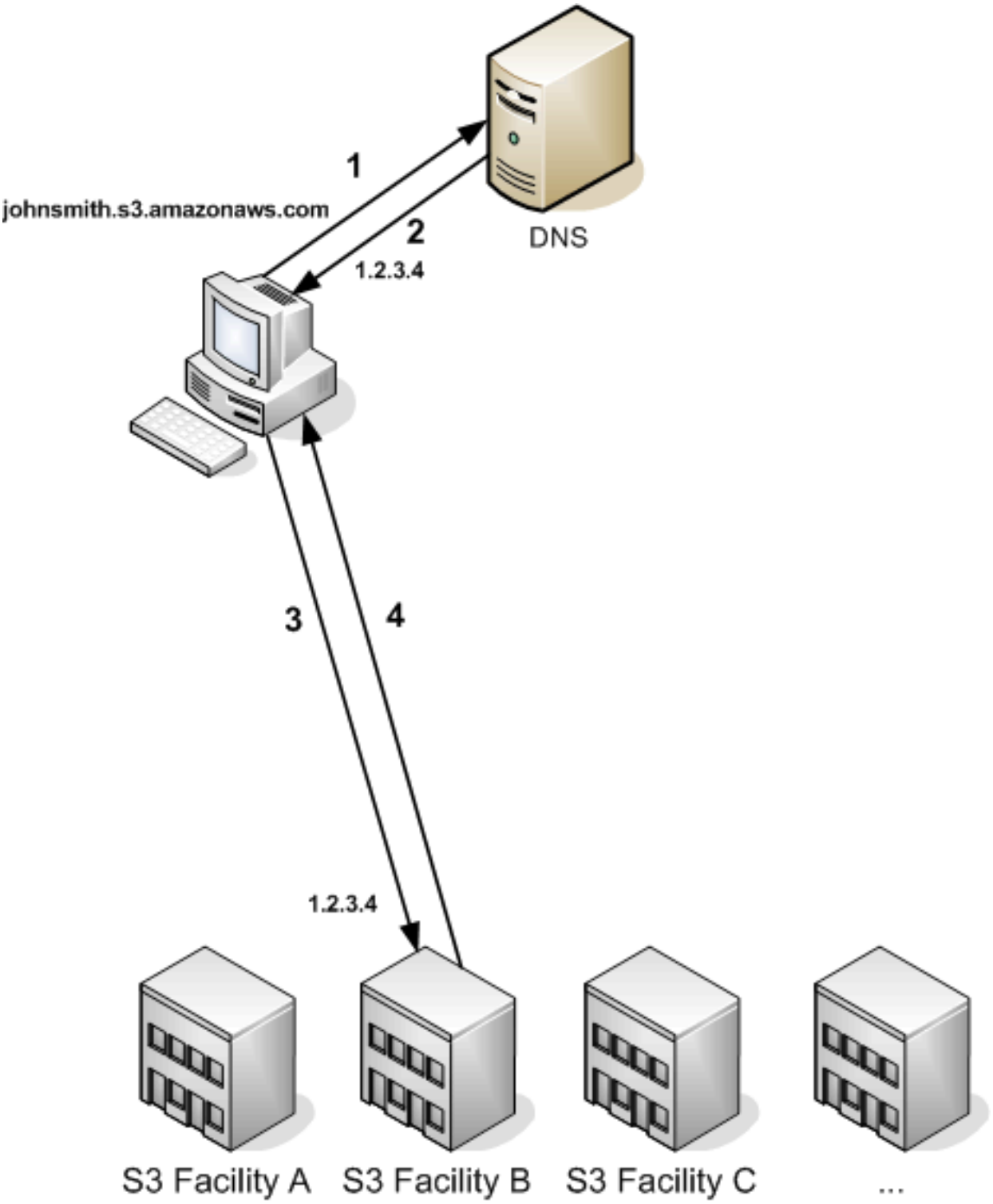
Untuk semua Wilayah yang diluncurkan setelah 20 Maret 2019, jika permintaan tiba di lokasi Amazon S3 yang salah, Amazon S3 mengembalikan kesalahan HTTP 400 Bad Request. Untuk informasi lebih lanjut tentang mengaktifkan atau menonaktifkanWilayah AWS, lihat Wilayah AWSdan Endpoint di Referensi Umum AWS.

Topik

- Rute DNS
- Pengalihan permintaan sementara
- Pengalihan permintaan permanen
- Meminta contoh pengalihan

Rute DNS

Rute perutean DNS meminta fasilitas Amazon S3 yang sesuai. Gambar dan prosedur berikut menunjukkan contoh perutean DNS.



Langkah permintaan perutean DNS

1. Klien mengajukan permintaan DNS untuk mendapatkan objek yang disimpan di Amazon S3.

2. Klien menerima satu atau lebih alamat IP untuk fasilitas yang dapat memproses permintaan. Dalam contoh ini, alamat IP adalah untuk Fasilitas B.
3. Klien meminta Amazon S3 Facility B.
4. Fasilitas B mengembalikan salinan objek kepada klien.

Pengalihan permintaan sementara

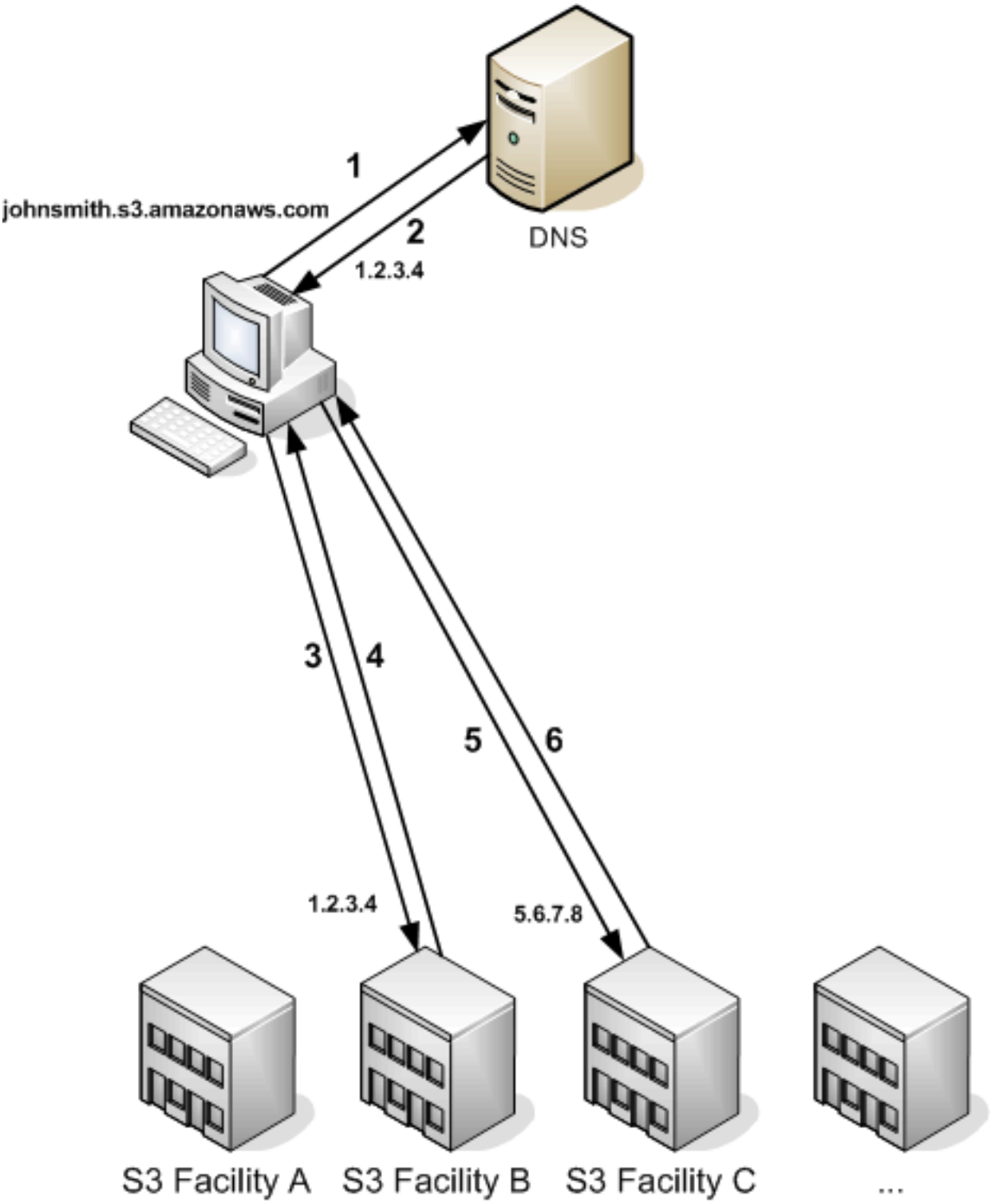
Pengalihan sementara adalah jenis respons kesalahan yang memberi sinyal kepada pemohon bahwa mereka harus mengirim ulang permintaan ke titik akhir yang berbeda. Karena sifat Amazon S3, permintaan dapat dikirimkan sementara ke fasilitas yang salah. Ini kemungkinan besar terjadi segera setelah bucket dibuat atau dihapus.

Misalnya, jika Anda membuat bucket baru dan segera membuat permintaan ke bucket, Anda mungkin menerima pengalihan sementara, tergantung pada kendala lokasi bucket. Jika Anda membuat bucket di Wilayah AWS US East (N. Virginia), Anda tidak akan melihat pengalihan karena ini juga merupakan titik akhir default Amazon S3.

Namun, jika bucket dibuat di Wilayah lain, setiap permintaan untuk bucket akan masuk ke titik akhir default sementara entri DNS kelompok dipropagasi. Titik akhir default mengarahkan kembali permintaan ke titik akhir yang benar dengan respons HTTP 302. Pengalihan sementara berisi URI ke fasilitas yang benar, yang dapat Anda gunakan untuk segera mengirim ulang permintaan.

> **Important**
>
> Jangan gunakan kembali titik akhir yang disediakan oleh respons pengalihan sebelumnya. Ini mungkin tampak berhasil (sekalipun untuk jangka waktu yang lama), tetapi dapat memberikan hasil yang tidak dapat diprediksi dan pada akhirnya akan gagal tanpa pemberitahuan.

Gambar dan prosedur berikut menunjukkan contoh pengalihan sementara.



Langkah pengalihan permintaan sementara

1. Klien mengajukan permintaan DNS untuk mendapatkan objek yang disimpan di Amazon S3.
2. Klien menerima satu atau lebih alamat IP untuk fasilitas yang dapat memproses permintaan.

3. Klien meminta Amazon S3 Facility B.
4. Fasilitas B mengembalikan pengalihan yang menunjukkan objek tersedia dari Lokasi C.
5. Klien mengirim ulang permintaan ke Fasilitas C.
6. Fasilitas C mengembalikan salinan objek.

Pengalihan permintaan permanen

Pengalihan permanen menunjukkan bahwa permintaan Anda menangani sumber daya secara tidak tepat. Misalnya, pengalihan permanen terjadi jika Anda menggunakan permintaan gaya jalur untuk mengakses bucket yang dibuat menggunakan `<CreateBucketConfiguration>`. Untuk informasi selengkapnya, lihat Mengakses dan mendaftarkan bucket Amazon S3.

Untuk membantu Anda menemukan kesalahan ini selama pengembangan, jenis pengalihan ini tidak berisi header HTTP Lokasi yang memungkinkan Anda secara otomatis mengikuti permintaan ke lokasi yang benar. Lihat dokumen kesalahan XML yang dihasilkan untuk bantuan menggunakan titik akhir Amazon S3.

Meminta contoh pengalihan

Berikut ini adalah contoh tanggapan pengalihan permintaan sementara.

REST API sementara mengalihkan respons

```
HTTP/1.1 307 Temporary Redirect
Location: http://awsexamplebucket1.s3-gztb4pa9sq.amazonaws.com/photos/puppy.jpg?
rk=e2c69a31
Content-Type: application/xml
Transfer-Encoding: chunked
Date: Fri, 12 Oct 2007 01:12:56 GMT
Server: AmazonS3

<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>TemporaryRedirect</Code>
  <Message>Please re-send this request to the specified temporary endpoint.
  Continue to use the original request endpoint for future requests.</Message>
  <Endpoint>awsexamplebucket1.s3-gztb4pa9sq.amazonaws.com</Endpoint>
</Error>
```

SOAP API sementara mengarahkan ulang respons

> **Note**
>
> Dukungan SOAP melalui HTTP dihentikan, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami merekomendasikan agar Anda menggunakan REST API atauAWS SDKs.

```
<soapenv:Body>
  <soapenv:Fault>
    <Faultcode>soapenv:Client.TemporaryRedirect</Faultcode>
    <Faultstring>Please re-send this request to the specified temporary endpoint.
    Continue to use the original request endpoint for future requests.</Faultstring>
    <Detail>
      <Bucket>images</Bucket>
      <Endpoint>s3-gztb4pa9sq.amazonaws.com</Endpoint>
    </Detail>
  </soapenv:Fault>
</soapenv:Body>
```

Pertimbangan DNS

Salah satu persyaratan desain Amazon S3 adalah ketersediaan yang sangat tinggi. Salah satu cara kami memenuhi persyaratan ini adalah dengan memperbarui alamat IP yang terkait dengan titik akhir Amazon S3 di DNS sesuai kebutuhan. Perubahan ini secara otomatis tercermin pada klien yang berumur pendek, tetapi tidak pada beberapa klien yang berumur panjang. Klien yang berumur panjang perlu mengambil tindakan khusus untuk menyelesaikan kembali titik akhir Amazon S3 secara berkala untuk mendapatkan keuntungan dari perubahan ini. Untuk informasi selengkapnya tentang mesin virtual (VM), lihat yang berikut ini:

- Untuk Java, JVM Sun menyimpan pencarian DNS selamanya secara default; buka bagian "InetAddressCaching” dari InetAddressdokumentasi untuk informasi tentang cara mengubah perilaku ini.
- Untuk PHP, VM PHP persisten yang berjalan di konfigurasi penerapan paling populer menyimpan cache pencarian DNS hingga VM dimulai ulang. Pergi ke getHostBynama PHP docs.

Menangani kesalahan REST dan SOAP

Topik

- Respons kesalahan REST
- Respons kesalahan SOAP
- Praktik terbaik kesalahan Amazon S3

Bagian ini menjelaskan kesalahan REST dan SOAP dan cara menanganinya.

> **Note**
>
> Dukungan SOAP melalui HTTP dihentikan, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami merekomendasikan agar Anda menggunakan REST API atauAWSSDK

Respons kesalahan REST

Jika permintaan REST mengakibatkan kesalahan, balasan HTTP memiliki:

- Dokumen kesalahan XML sebagai bodi respons
- Tipe Konten: aplikasi/xml
- 3xx, 4xx, atau 5xx kode status HTTP yang sesuai

Berikut ini adalah contoh Respons Kesalahan REST.

```
<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>NoSuchKey</Code>
  <Message>The resource you requested does not exist</Message>
  <Resource>/mybucket/myfoto.jpg</Resource>
  <RequestId>4442587FB7D0A2F9</RequestId>
</Error>
```

Untuk informasi selengkapnya tentang kesalahan Amazon S3, kunjungi ErrorCodeList.

Header respons

Berikut ini adalah header respons yang dikembalikan oleh semua operasi:

- `x-amz-request-id:` ID unik yang ditetapkan untuk setiap permintaan oleh sistem. Jika Anda memiliki masalah dengan Amazon S3, Amazon dapat menggunakannya untuk membantu memecahkan masalah.
- `x-amz-id-2:` Token khusus yang akan membantu kami memecahkan masalah.

Respons kesalahan

Saat permintaan Amazon S3 mengalami kesalahan, klien menerima tanggapan kesalahan. Format yang tepat dari respons kesalahan adalah spesifik API: Misalnya, respons kesalahan REST berbeda dari respons kesalahan SOAP. Namun, semua respons kesalahan memiliki elemen umum.

> **Note**
>
> Dukungan SOAP melalui HTTP dihentikan, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami merekomendasikan agar Anda menggunakan REST API atauAWSSDK

Kode kesalahan

Kode kesalahan adalah string yang secara unik mengidentifikasi kondisi kesalahan. Ini dimaksudkan untuk dibaca dan dipahami oleh program yang mendeteksi dan menangani kesalahan berdasarkan tipe. Banyak kode kesalahan yang umum di seluruh SOAP dan REST , tetapi beberapa di antaranya adalah API yang bersifat spesifik. Misalnya, NoSuchKey bersifat universal, tapi UnexpectedContent dapat terjadi sebagai tanggapan atas permintaan REST yang tidak valid. Dalam semua kasus, kode kesalahan SOAP membawa awalan seperti yang ditunjukkan dalam tabel kode kesalahan, sehingga NoSuchKey kesalahan sebenarnya dikembalikan dalam SOAP sebagai Client.NoSuchKey.

> **Note**
>
> Dukungan SOAP melalui HTTP dihentikan, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami merekomendasikan agar Anda menggunakan REST API atauAWSSDK

Pesan kesalahan

Pesan kesalahan berisi deskripsi umum kondisi kesalahan dalam bahasa Inggris. Ini ditujukan untuk audiens manusia. Program sederhana menampilkan pesan secara langsung kepada pengguna akhir jika mereka mengalami kondisi kesalahan yang tidak mereka ketahui atau tidak peduli untuk menanganinya. Program-program canggih dengan penanganan kesalahan yang lebih lengkap dan internasionalisasi yang tepat lebih cenderung mengabaikan pesan kesalahan.

Detail lebih lanjut

Banyak respons kesalahan berisi data terstruktur tambahan yang dimaksudkan untuk dibaca dan dipahami oleh pengembang yang mendiagnosis kesalahan pemrograman. Misalnya, jika Anda mengirim header Content-MD5 dengan permintaan REST PUT yang tidak cocok dengan ringkasan yang dihitung di server, Anda menerima BadDigest kesalahan. Respons kesalahan juga menyertakan sebagai elemen detail digest yang kami hitung, dan digest yang Anda beri tahukan kepada kami. Selama pengembangan, Anda dapat menggunakan informasi ini untuk mendiagnosis kesalahan. Dalam produksi, program yang berperilaku baik mungkin menyertakan informasi ini dalam log kesalahannya.

Respons kesalahan SOAP

> **Note**
>
> Dukungan SOAP melalui HTTP dihentikan, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami merekomendasikan agar Anda menggunakan REST API atauAWSSDK

Dalam SOAP, hasil kesalahan dikembalikan kepada klien sebagai kesalahan SOAP, dengan kode respons HTTP 500. Jika Anda tidak menerima kesalahan SOAP, maka permintaan Anda berhasil. Kode kesalahan SOAP Amazon S3 terdiri atas kode kesalahan SOAP 1.1 standar (baik "Server" atau "Client") yang digabungkan dengan kode kesalahan spesifik Amazon S3. Misalnya: “Server.InternalError“atau “Klien.NoSuchBucket“. Elemen string kesalahan SOAP berisi pesan kesalahan umum yang dapat dibaca oleh manusia dalam bahasa Inggris. Terakhir, elemen detail kesalahan SOAP mengandung berbagai informasi yang relevan dengan kesalahan.

Misalnya, jika Anda mencoba menghapus objek “Fred”, yang tidak ada, bodi jawaban SOAP berisi”NoSuchKey“Kesalahan SOAP.

Example

```
<soapenv:Body>
  <soapenv:Fault>
    <Faultcode>soapenv:Client.NoSuchKey</Faultcode>
    <Faultstring>The specified key does not exist.</Faultstring>
    <Detail>
      <Key>Fred</Key>
    </Detail>
  </soapenv:Fault>
</soapenv:Body>
```

Untuk informasi selengkapnya tentang kesalahan Amazon S3, kunjungi ErrorCodeList.

Praktik terbaik kesalahan Amazon S3

Saat merancang aplikasi untuk digunakan dengan Amazon S3, penting untuk menangani kesalahan Amazon S3 sebagaimana mestinya. Bagian ini menjelaskan masalah yang perlu dipertimbangkan saat merancang aplikasi Anda.

Coba lagi InternalErrors

Kesalahan internal adalah kesalahan yang terjadi dalam lingkungan Amazon S3.

Permintaan yang menerima InternalError respons mungkin belum diproses. Misalnya, jika permintaan PUT kembali InternalError, GET berikutnya mungkin mengambil nilai lama atau nilai yang diperbarui.

Jika Amazon S3 mengembalikan InternalError respons, coba lagi permintaannya.

Sesuaikan aplikasi untuk kesalahan SlowDown yang berulang

Seperti seperti sistem terdistribusi lainnya, S3 memiliki mekanisme perlindungan yang mendeteksi konsumsi berlebihan sumber daya yang disengaja atau tidak disengaja dan bereaksi sesuai itu. SlowDown kesalahan dapat terjadi ketika tingkat permintaan yang tinggi memicu salah satu mekanisme ini. Mengurangi tingkat permintaan Anda akan mengurangi atau menghilangkan kesalahan tipe ini. Secara umum, sebagian besar pengguna tidak akan mengalami kesalahan ini secara teratur; namun, jika Anda menginginkan informasi lebih lanjut atau mengalami kesalahan yang tinggi atau tidak terduga SlowDown kesalahan, silahkan posting ke kamiForum pengembang Amazon S3atau mendaftar keAWS Support https://aws.amazon.com/premiumsupport/.

Isolasi kesalahan

> **Note**
>
> Dukungan SOAP melalui HTTP dihentikan, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami merekomendasikan agar Anda menggunakan REST API atauAWSSDK

Amazon S3 menyediakan serangkaian kode kesalahan yang digunakan oleh API SOAP dan REST. API SOAP mengembalikan kode kesalahan Amazon S3 standar. API REST dirancang untuk terlihat seperti server HTTP standar dan berinteraksi dengan klien HTTP yang ada (mis. peramban, pustaka klien HTTP, proksi, cache, dan lain-lain). Untuk memastikan klien HTTP menangani kesalahan dengan benar, kami memetakan setiap kesalahan Amazon S3 ke kode status HTTP.

Kode status HTTP kurang ekspresif dibandingkan kode kesalahan Amazon S3 dan berisi lebih sedikit informasi tentang kesalahan tersebut. Misalnya, kesalahan `NoSuchKey` dan `NoSuchBucket` Amazon S3 keduanya memetakan ke kode status `HTTP 404 Not Found`.

Meskipun kode status HTTP memuat lebih sedikit informasi tentang kesalahan, klien yang memahami HTTP, tetapi bukan API Amazon S3, biasanya akan menangani kesalahan dengan benar.

Oleh karena itu, saat menangani kesalahan atau melaporkan kesalahan Amazon S3 kepada pengguna akhir, gunakan kode kesalahan Amazon S3 alih-alih kode status HTTP karena kode ini berisi sebagian besar informasi tentang kesalahan tersebut. Selain itu, saat men-debug aplikasi Anda, Anda juga harus melihat elemen <Details> yang dapat dibaca manusia dari respons kesalahan XML.

Referensi developer

Lampiran ini berisi bagian-bagian berikut.

Topik

- Lampiran a: Menggunakan SOAP API
- Lampiran b: Mengautentikasi permintaan (versi AWS tanda tangan 2)

Lampiran a: Menggunakan SOAP API

> **Note**
>
> Dukungan SOAP melalui HTTP tidak digunakan lagi, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami sarankan Anda menggunakan REST API atau AWS SDK.

Bagian ini berisi informasi khusus tentang Amazon S3 SOAP API.

> **Note**
>
> Permintaan SOAP, baik yang diautentikasi maupun yang anonim, harus dikirim ke Amazon S3 menggunakan SSL. Amazon S3 akan menampilkan pesan kesalahan saat Anda mengirim permintaan SOAP melalui HTTP.

Topik

- Elemen-elemen umum SOAP API
- Mengautentikasi permintaan SOAP
- Mengatur kebijakan akses dengan SOAP

Elemen-elemen umum SOAP API

> **Note**
>
> Dukungan SOAP melalui HTTP dihentikan, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami merekomendasikan agar Anda menggunakan REST API atauAWSSDK.

Anda dapat berinteraksi dengan Amazon S3 menggunakan SOAP 1.1 melalui HTTP. Amazon S3 WSDL, yang menjelaskan Amazon S3 API dengan cara yang dapat dibaca mesin, tersedia di: https://doc.s3.amazonaws.com/2006-03-01/AmazonS3.wsdl. Skema Amazon S3 tersedia di https://doc.s3.amazonaws.com/2006-03-01/AmazonS3.xsd.

Sebagian besar pengguna akan berinteraksi dengan Amazon S3 menggunakan toolkit SOAP yang disesuaikan untuk bahasa dan lingkungan pengembangan mereka. Toolkit yang berbeda akan menampilkan Amazon S3 API dengan cara yang berbeda. Bacalah dokumentasi toolkit khusus Anda untuk memahami cara menggunakannya. Bagian ini menggambarkan operasi SOAP Amazon S3 secara mandiri dengan menunjukkan permintaan dan respons XML saat muncul “on the wire.”

Elemen-elemen umum

Anda dapat menyertakan elemen-elemen terkait otorisasi berikut dengan permintaan SOAP:

- `AWSAccessKeyId:` Access Key ID AWS dari peminta
- `Timestamp:` Waktu saat ini pada sistem Anda
- `Signature:` Tanda tangan untuk permintaan

Mengautentikasi permintaan SOAP

> **Note**
>
> Dukungan SOAP melalui HTTP dihentikan, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami menyarankan agar Anda menggunakan REST API atauAWSSDK.

Setiap permintaan non-anonim harus berisi informasi autentikasi untuk menetapkan identitas prinsipal yang membuat permintaan tersebut. Dalam SOAP, informasi autentikasi dimasukkan ke dalam elemen-elemen permintaan SOAP berikut:

- Access Key ID AWS Anda

  > **Note**
  >
  > Saat membuat permintaan SOAP yang diautentikasi, kredensial keamanan sementara tidak didukung. Untuk informasi selengkapnya tentang jenis kredensial, lihat Membuat permintaan.

- `Timestamp:` Ini harus sebuah dateTime (masuk ke http://www.w3.org/TR/xmlschema-2/#dateTime) di zona waktu Coordinated Universal Time (Greenwich Mean Time), seperti

`2009-01-01T12:00:00.000Z`. Otorisasi akan gagal jika timestamp ini lebih dari 15 menit dari jam pada server Amazon S3.

- `Signature:` RFC 2104 HMAC-SHA1 Digest (Buka http://www.ietf.org/rfc/rfc2104.txt) dari gabungan "Amazon3" + OPERASI + Timestamp, menggunakan Secret Access Key AWS Anda sebagai kuncinya. Misalnya, dalam yang berikut ini CreateBucket permintaan sampel, elemen tanda tangan akan mengandung intisari HMAC-SHA1 dari nilai “AmazonS3CreateBucket2009-01-01T 12:00:00 .000Z”:

Misalnya, dalam yang berikut ini CreateBucket permintaan sampel, elemen tanda tangan akan mengandung intisari HMAC-SHA1 dari nilai “AmazonS3CreateBucket2009-01-01T 12:00:00 .000Z”:

Example

```
<CreateBucket xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Acl>private</Acl>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2009-01-01T12:00:00.000Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</CreateBucket>
```

> **Note**
>
> Permintaan SOAP, baik yang diautentikasi maupun anonim, harus dikirim ke Amazon S3 menggunakan SSL. Amazon S3 mengembalikan kesalahan saat Anda mengirimkan permintaan SOAP melalui HTTP.

> **Important**
>
> Karena perbedaan interpretasi tentang bagaimana presisi waktu ekstra harus ditempatkan, pengguna .NET harus berhati-hati agar tidak mengirimkan stempel waktu Amazon S3 secara berlebihan. Ini dapat dicapai dengan menyusun `DateTime` objek secara manual dengan presisi hanya dalam milidetik.

Mengatur kebijakan akses dengan SOAP

> **Note**
>
> Dukungan SOAP melalui HTTP tidak digunakan lagi, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami sarankan Anda menggunakan REST API atau AWS SDK.

Kontrol akses dapat diatur pada saat ember atau objek ditulis dengan menyertakan elemen "AccessControlList" dengan permintaan untuk`CreateBucket`,`PutObjectInline`, atau`PutObject`. AccessControlList Elemen tersebut dijelaskan dalamIdentity and Access Management untuk Amazon S3. Jika tidak ada daftar kontrol akses yang ditentukan dengan operasi ini, sumber daya dibuat dengan kebijakan akses default yang memberikan akses FULL_CONTROL pemohon (ini adalah kasus bahkan jika permintaan adalah PutObjectInline atau PutObject permintaan untuk objek yang sudah ada).

Berikut ini adalah permintaan yang menulis data ke sebuah objek, membuat objek tersebut dapat dibaca oleh pengguna utama anonim, dan memberi pengguna tersebut hak-hak FULL_CONTROL ke bucket (Sebagian besar pengembang akan memberikan akses FULL_CONTROL ke bucket mereka sendiri).

Example

Berikut ini adalah permintaan yang menulis data ke sebuah objek dan membuat objek tersebut dapat dibaca oleh pengguna utama anonim.

`Sample Request`

```
<PutObjectInline xmlns="https://doc.s3.amazonaws.com/2006-03-01">
  <Bucket>quotes</Bucket>
  <Key>Nelson</Key>
  <Metadata>
    <Name>Content-Type</Name>
    <Value>text/plain</Value>
  </Metadata>
  <Data>aGEtaGE=</Data>
  <ContentLength>5</ContentLength>
  <AccessControlList>
    <Grant>
```

```
      <Grantee xsi:type="CanonicalUser">
        <ID>75cc57f09aa0c8caeab4f8c24e99d10f8e7faeebf76c078efc7c6caea54ba06a</ID>
        <DisplayName>chriscustomer</DisplayName>
      </Grantee>
      <Permission>FULL_CONTROL</Permission>
    </Grant>
    <Grant>
      <Grantee xsi:type="Group">
        <URI>http://acs.amazonaws.com/groups/global/AllUsers<URI>
      </Grantee>
      <Permission>READ</Permission>
    </Grant>
  </AccessControlList>
  <AWSAccessKeyId>AKIAIOSFODNN7EXAMPLE</AWSAccessKeyId>
  <Timestamp>2009-03-01T12:00:00.183Z</Timestamp>
  <Signature>Iuyz3d3P0aTou39dzbqaEXAMPLE=</Signature>
</PutObjectInline>
```

Sample Response

```
<PutObjectInlineResponse xmlns="https://s3.amazonaws.com/doc/2006-03-01">
  <PutObjectInlineResponse>
    <ETag>&quot828ef3fdfa96f00ad9f27c383fc9ac7f&quot</ETag>
    <LastModified>2009-01-01T12:00:00.000Z</LastModified>
  </PutObjectInlineResponse>
</PutObjectInlineResponse>
```

Kebijakan kontrol akses dapat dibaca atau diatur untuk bucket atau objek yang sudah ada dengan menggunakan metode `GetBucketAccessControlPolicy`, `GetObjectAccessControlPolicy`, `SetBucketAccessControlPolicy`, dan `SetObjectAccessControlPolicy`. Untuk informasi lebih lanjut, lihat penjelasan terperinci tentang metode-metode tersebut.

Lampiran b: Mengautentikasi permintaan (versi AWS tanda tangan 2)

> **Important**
>
> Bagian ini menjelaskan cara mengautentikasi permintaan menggunakan AWS Signature Version 2. Tanda Tangan Versi 2 akan dinonaktifkan (dihilangkan), Amazon S3 hanya akan menerima permintaan API yang ditandatangani menggunakan Tanda Tangan Versi 4. Untuk informasi selengkapnya, lihat AWS Tanda Tangan Versi 2 Dimatikan (Tidak Digunakan Lagi) untuk Amazon S3

Signature Version 4 didukung di semua Wilayah AWS, dan ini adalah satu-satunya versi yang didukung untuk Wilayah baru. Untuk informasi selengkapnya, lihat Mengautentikasi Permintaan (Versi AWS Tanda Tangan 4) di Referensi API Amazon Simple Storage Service. Amazon S3 menawarkan kemampuan mengidentifikasi versi tanda tangan API yang digunakan untuk menandatangani permintaan. Penting untuk mengidentifikasi apakah salah satu alur kerja Anda menggunakan penandatanganan dengan Tanda Tangan Versi 2 dan meningkatkannya menjadi Tanda Tangan Versi 4 untuk mencegah dampaknya terhadap bisnis Anda.

- Jika Anda menggunakan log CloudTrail peristiwa (opsi yang disarankan), silakan lihat Mengidentifikasi permintaan Amazon S3 Signature Version 2 dengan menggunakan CloudTrail cara menanyakan dan mengidentifikasi permintaan tersebut.
- Jika Anda menggunakan log Akses Server Amazon S3, lihat Mengidentifikasi permintaan Signature Version 2 menggunakan pencatatan akses Amazon S3

Topik

- Mengautentikasi permintaan menggunakan API REST
- Menandatangani dan mengautentikasi permintaan REST
- Unggahan berbasis browser menggunakan POST (versi AWS tanda tangan 2)

Mengautentikasi permintaan menggunakan API REST

Saat mengakses Amazon S3 menggunakan REST, Anda harus menyediakan item-item berikut dalam permintaan Anda agar permintaan tersebut dapat diautentikasi:

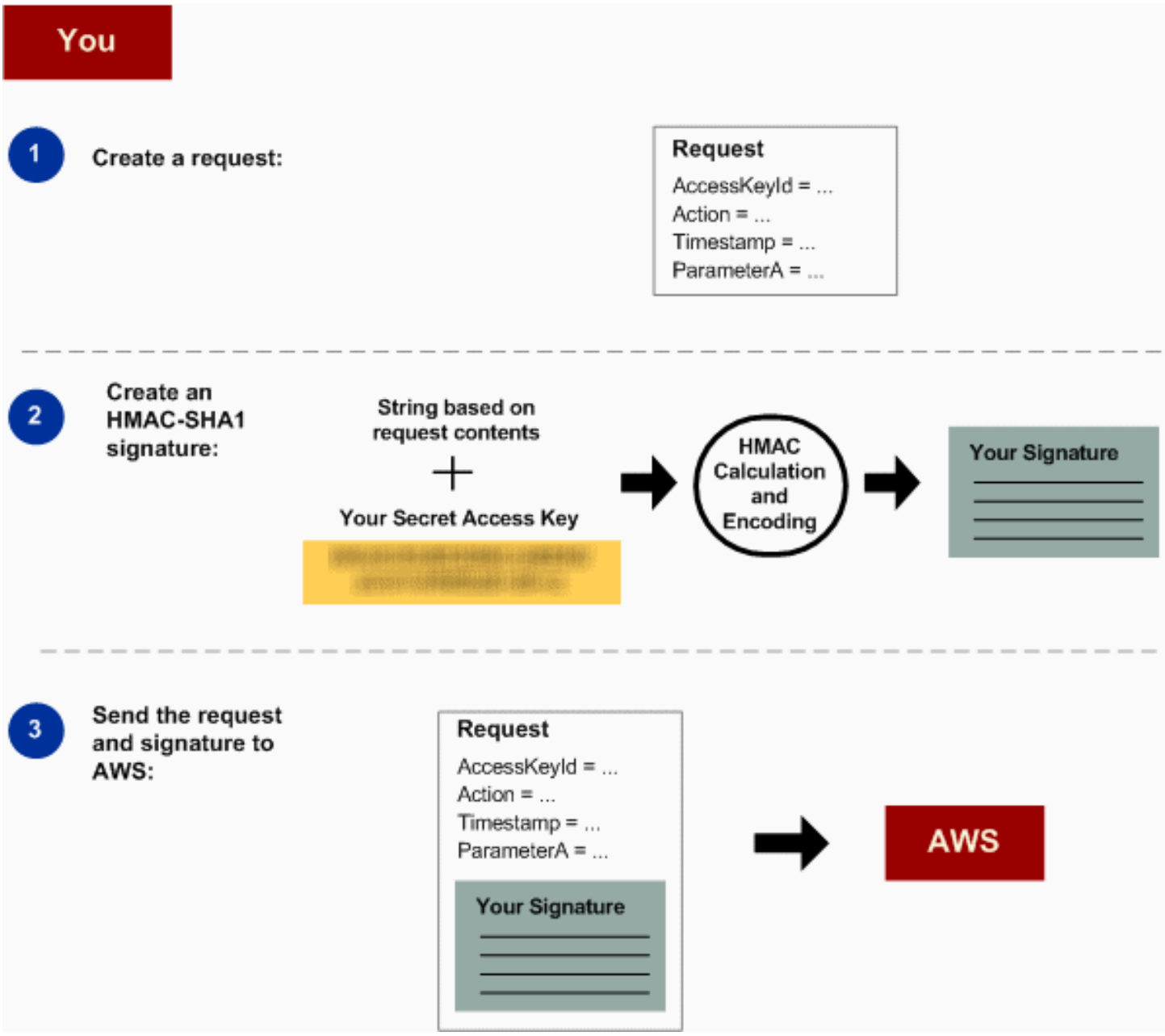
Elemen permintaan

- AWS ID kunci akses - Setiap permintaan harus berisi ID kunci akses dari identitas yang Anda gunakan untuk mengirim permintaan Anda.
- Tanda Tangan–Setiap permintaan harus berisi tanda tangan permintaan yang valid, jika tidak, permintaan akan ditolak.

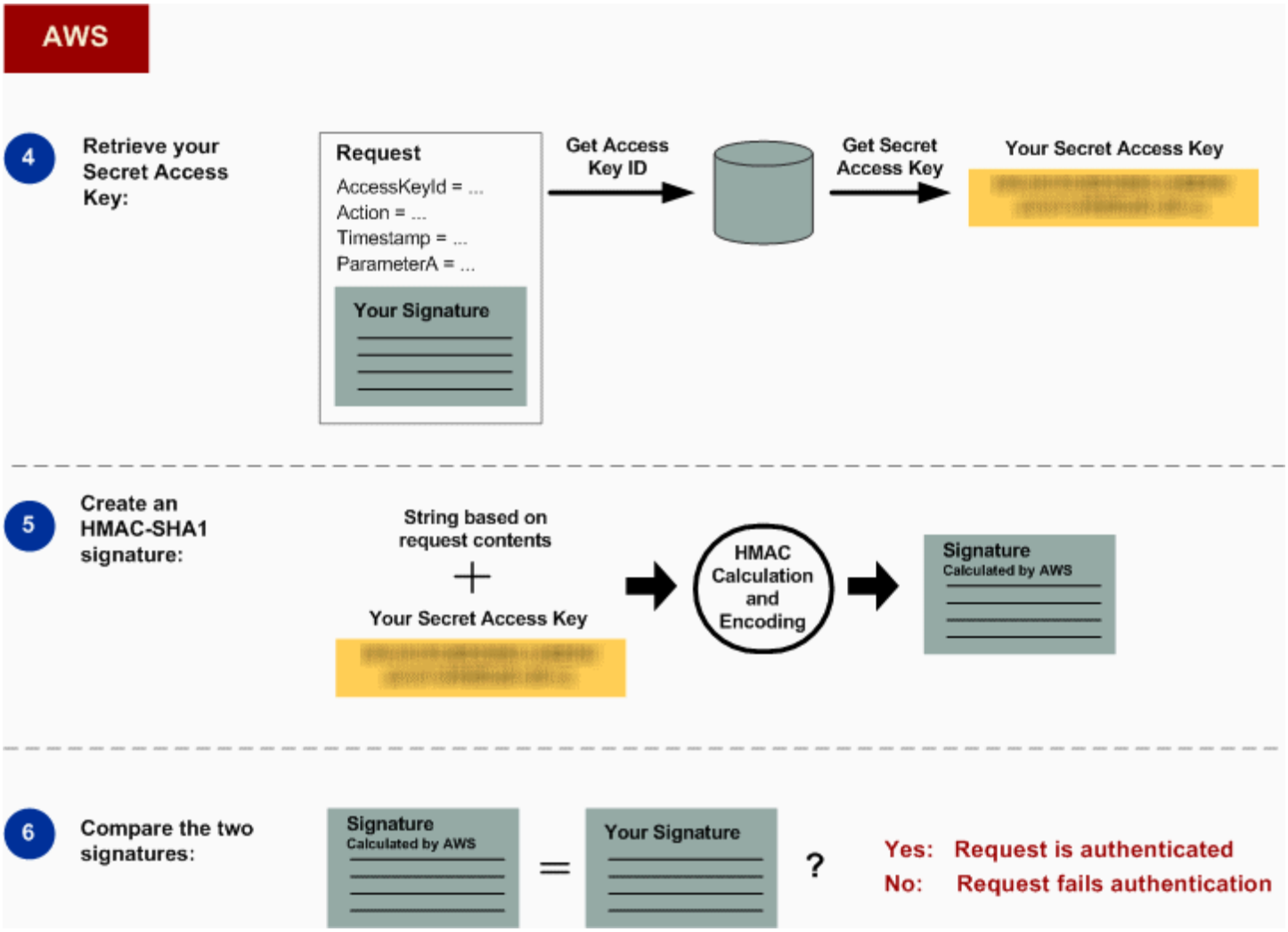
  Tanda tangan permintaan dihitung menggunakan kunci akses rahasia, yang merupakan rahasia bersama yang hanya diketahui oleh Anda dan AWS.
- Stempel waktu–Setiap permintaan harus berisi tanggal dan waktu pembuatan permintaan yang direpresentasikan sebagai string dalam UTC.
- Tanggal–Setiap permintaan harus mencantumkan stempel waktu permintaan.

  Tergantung pada tindakan API yang digunakan, Anda bisa memberikan tanggal dan waktu kedaluwarsa untuk permintaan, sebagai pengganti atau sebagai tambahan dari stempel waktu. Lihat topik autentikasi untuk tindakan tertentu agar dapat menentukan hal yang diperlukan.

Berikut adalah langkah umum untuk mengautentikasi permintaan ke Amazon S3. Anggaplah bahwa Anda telah memiliki kredensial keamanan, ID kunci akses, dan kunci akses rahasia yang diperlukan.



1	Membangun permintaan untuk AWS.
2	Hitung tanda tangan menggunakan kunci akses rahasia Anda.
3	Kirim permintaan ke Amazon S3. Sertakan ID kunci akses dan tanda tangan dalam permintaan Anda. Amazon S3 melakukan tiga langkah berikutnya.



4	Amazon S3 menggunakan ID kunci akses untuk mencari kunci akses rahasia Anda.
5	Amazon S3 menghitung tanda tangan dari data permintaan dan kunci akses rahasia menggunakan algoritma yang sama dengan yang Anda gunakan untuk menghitung tanda tangan yang dikirim dalam permintaan.
6	Jika tanda tangan yang dihasilkan oleh Amazon S3 sesuai dengan yang Anda kirim dalam permintaan, permintaan tersebut dianggap benar. Jika perbandingan gagal, permintaan dibatalkan, dan Amazon S3 akan memberikan respons kesalahan.

Informasi autentikasi terperinci

Untuk informasi terperinci tentang autentikasi REST, lihat Menandatangani dan mengautentikasi permintaan REST.

Menandatangani dan mengautentikasi permintaan REST

Topik

- Menggunakan kredensial keamanan sementara
- Header autentikasi
- Kanonikalisasi permintaan untuk penandatanganan
- Membuat CanonicalizedResource elemen
- Membuat CanonicalizedAmzHeaders elemen
- ELemen header posisi versus elemen StringToSign header HTTP yang diberi nama
- Ketentuan stempel waktu
- Contoh-contoh autentikasi
- Masalah penandatanganan permintaan REST
- Alternatif autentikasi permintaan string kueri

> **Note**
>
> Topik ini menjelaskan tentang autentikasi permintaan dengan menggunakan Tanda Tangan Versi 2. Amazon S3 kini mendukung Tanda Tangan Versi 4 terbaru. Versi tanda tangan terbaru ini didukung di semua wilayah dan setiap wilayah baru setelah 30 Januari 2014 hanya akan mendukung Tanda Tangan Versi 4. Untuk informasi selengkapnya, kunjungi Permintaan Autentikasi (Tanda Tangan Versi 4 AWS) di Referensi API Amazon Simple Storage Service.

Autentikasi adalah proses pembuktian identitas Anda pada sistem. Identitas adalah faktor penting dalam keputusan kontrol akses Amazon S3. Permintaan diizinkan atau ditolak sebagian berdasarkan identitas pemohon. Sebagai contoh, hak untuk membuat bucket disediakan untuk pengembang yang terdaftar dan (secara default) hak untuk membuat objek di dalam bucket disediakan untuk pemilik bucket yang bersangkutan. Sebagai pengembang, Anda akan membuat permintaan yang menginvokasi hak istimewa ini, jadi Anda perlu membuktikan identitas pada sistem dengan melakukan autentikasi atas permintaan Anda. Dalam bagian ini, akan ditunjukkan caranya.

> **Note**
>
> Konten dalam bagian ini tidak berlaku untuk POST HTTP. Untuk informasi selengkapnya, lihat Unggahan berbasis browser menggunakan POST (versi AWS tanda tangan 2).

API REST Amazon S3 menggunakan skema HTTP kustom berdasarkan kunci-HMAC (Kode Autentikasi Pesan Hash) untuk autentikasi. Untuk mengautentikasi permintaan, pertama-tama Anda perlu menggabungkan elemen-elemen yang dipilih dari permintaan tersebut untuk membentuk string. Kemudian, gunakan kunci akses rahasia AWS untuk menghitung HMAC dari string tersebut. Secara informal, kami menyebut proses ini sebagai "penandatanganan permintaan," dan kami menyebut output dari algoritma HMAC sebagai tanda tangan, karena proses ini mensimulasikan properti keamanan dari tanda tangan yang sebenarnya. Terakhir, Anda perlu menambahkan tanda tangan ini sebagai parameter permintaan dengan menggunakan sintaks yang dijelaskan dalam bagian ini.

Saat sistem menerima permintaan yang telah diautentikasi, sistem akan mengambil kunci akses rahasia AWS yang Anda klaim dan menggunakannya dengan cara yang sama untuk menghitung tanda tangan untuk pesan yang diterima. Kemudian, sistem akan membandingkan tanda tangan yang dihitung dengan tanda tangan yang diberikan oleh pemohon. Jika kedua tanda tangan cocok, maka sistem akan menyimpulkan bahwa peminta harus memiliki akses ke kunci akses rahasia AWS dan oleh karena itu sistem bertindak dengan otoritas pengguna utama yang diberikan kunci tersebut. Jika kedua tanda tangan tidak cocok, permintaan akan dibatalkan dan sistem merespons dengan mengirimkan pesan kesalahan.

Example Permintaan REST Amazon S3 yang telah diautentikasi

```
GET /photos/puppy.jpg HTTP/1.1
Host: awsexamplebucket1.us-west-1.s3.amazonaws.com
Date: Tue, 27 Mar 2007 19:36:42 +0000

Authorization: AWS AKIAIOSFODNN7EXAMPLE:
qgk2+6Sv9/oM7G3qLEjTH1a1l1g=
```

Menggunakan kredensial keamanan sementara

Jika Anda menandatangani permintaan menggunakan kredensial keamanan sementara (lihat Membuat permintaan), Anda harus menyertakan token keamanan yang sesuai dalam permintaan dengan menambahkan header `x-amz-security-token`.

Saat Anda memperoleh kredensial keamanan sementara menggunakan API AWS Security Token Service, respons tersebut mencakup kredensial keamanan sementara dan token sesi. Anda memberikan nilai token sesi dalam header `x-amz-security-token` saat Anda mengirim permintaan ke Amazon S3. Untuk informasi tentang API AWS Security Token Service yang disediakan oleh IAM, masuk ke Tindakan di AWS Security Token Service Panduan Referensi API .

Header autentikasi

API REST Amazon S3 menggunakan header HTTP `Authorization` standar untuk memberikan informasi autentikasi. (Nama header standar tidak disarankan karena memuat informasi autentikasi, bukan otorisasi.) Berdasarkan skema autentikasi Amazon S3, header Otorisasi memiliki bentuk berikut ini:

```
Authorization: AWS AWSAccessKeyId:Signature
```

Pegembang menerbitkan kunci akses ID AWS dan kunci akses rahasia AWS saat mereka mendaftar. Untuk autentikasi permintaan, elemen `AWSAccessKeyId` mengidentifikasi ID kunci akses yang digunakan untuk menghitung tanda tangan dan secara tidak langsung, juga mengidentifikasi pengembang yang mengajukan permintaan.

Elemen `Signature` adalah RFC 2104 HMAC-SHA1 dari elemen-elemen yang dipilih dari permintaan, sehingga bagian `Signature` header Otorisasi akan berbeda-beda untuk setiap permintaan. Jika tanda tangan permintaan yang dihitung oleh sistem sesuai dengan `Signature` yang disertakan dalam permintaan, maka peminta akan menunjukkan kepemilikan kunci akses rahasia AWS. Permintaan tersebut kemudian akan diproses menggunakan identitas tersebut dan dengan otoritas dari pengembang yang diberikan kunci tersebut.

Berikut ini adalah pseudogrammar yang mengilustrasikan konstruksi header permintaan `Authorization`. (Dalam contoh tersebut, `\n` berarti titik kode Unicode `U+000A`, umumnya disebut sebagai baris baru).

```
Authorization = "AWS" + " " + AWSAccessKeyId + ":" + Signature;

Signature = Base64( HMAC-SHA1( UTF-8-Encoding-Of(YourSecretAccessKey), UTF-8-Encoding-
Of( StringToSign ) ) );

StringToSign = HTTP-Verb + "\n" +
 Content-MD5 + "\n" +
```

```
 Content-Type + "\n" +
 Date + "\n" +
 CanonicalizedAmzHeaders +
 CanonicalizedResource;

CanonicalizedResource = [ "/" + Bucket ] +
 <HTTP-Request-URI, from the protocol name up to the query string> +
 [ subresource, if present. For example "?acl", "?location", or "?logging"];

CanonicalizedAmzHeaders = <described below>
```

HMAC-SHA1 adalah algoritma yang ditentukan oleh RFC 2104 - Keyed-Hashing untuk Autentikasi Pesan . Algoritma tersebut menerima dua string byte sebagai input, yaitu kunci dan pesan. Untuk autentikasi permintaan Amazon S3, gunakan kunci akses rahasia AWS (`YourSecretAccessKey`) sebagai kunci, dan pengkodean UTF-8 `StringToSign` sebagai pesan. Output HMAC-SHA1 juga merupakan string byte, yang disebut digest. Parameter permintaan `Signature` dibangun berdasarkan Base64 yang mengkodekan digest ini.

Kanonikalisasi permintaan untuk penandatanganan

Ingatlah bahwa ketika sistem menerima permintaan yang diautentikasi, sistem tersebut membandingkan tanda tangan permintaan yang dikomputasi dengan tanda tangan yang disertakan dalam permintaan di `StringToSign`. Oleh karena itu, Anda harus melakukan komputasi tanda tangan dengan menggunakan metode yang sama seperti yang digunakan oleh Amazon S3. Kami menyebut proses permohonan dalam bentuk yang telah disetujui untuk penandatanganan sebagai kanonikalisasi.

Membuat CanonicalizedResource elemen

`CanonicalizedResource` mewakili sumber daya Amazon S3 yang ditarget oleh permintaan. Buat elemen permintaan REST sebagai berikut:

Proses peluncuran

1	Mulai dengan string kosong (`""`).
2	Jika permintaan menetapkan bucket untuk menggunakan header Host HTTP (gaya hosting virtual), tambahkan nama bucket yang diawali dengan `"/"` (misalnya, "/bucketname"). Untuk permintaan yang menggunakan gaya penulisan path dan permintaan yang tidak ditujukan ke bucket, jangan lakukan apa pun. Untuk informasi selengkapnya tentang permintaan cara hosting virtual, lihat Bucket dengan hosting virtual.

	Untuk permintaan gaya hosting virtual "https://awsexamplebucket1.s3.us-west-1.amazonaws.com/photos/puppy.jpg", `CanonicalizedResource` adalah "/awsexamplebucket1". Untuk permintaan dengan gaya penulisan path, “https://s3.us-west-1.amazonaws.com/awsexamplebucket1/photos/puppy.jpg", `CanonicalizedResource` adalah "".
3	Tambahkan bagian alur dari URI Permintaan HTTP yang tidak di-decoding, hingga tetapi tidak termasuk string kueri. Untuk permintaan gaya penulisan hosting virtual "https://awsexamplebucket1.s3.us-west-1.amazonaws.com/photos/puppy.jpg", `CanonicalizedResource` adalah “/awsexamplebucket1/photos/puppy.jpg”. Untuk permintaan gaya penulisan path, "https://s3.us-west-1.amazonaws.com/awsexamplebucket1/photos/puppy.jpg", `CanonicalizedResource` adalah “/awsexamplebucket1/photos/puppy.jpg”. Pada titik ini, `CanonicalizedResource` sama, baik untuk permintaan gaya penulisan hosting virtual dan permintaan gaya penulisan path. Untuk permintaan yang tidak menyebutkan bucket, misalnya Layanan DAPATKAN, tambahkan "/".

4	Jika permintaan membahas sub-sumber daya, seperti `?versioning` , `?location` , `?acl`, `?lifecycle` , atau `?versionid` tambahkan sub-sumber dayanya, nilainya, jika ada, dan tanda tanya. Perhatikan bahwa jika terdapat beberapa sub-sumber daya, maka sub-sumber daya tersebut harus diurutkan secara leksikografis berdasarkan nama sub-sumber daya dan dipisahkan oleh '&', misalnya, ?acl&versionId=*`nilai`*. Subsumber daya yang harus disertakan saat membuat CanonicalizedResource Elemen adalah acl, lifecycle, location, logging, notification, partNumber, policy, requestPayment, uploadid, upload, versionId, versionId, versioning, version, dan website. Jika permintaan menentukan parameter string kueri yang menimpa nilai header respons (lihat Dapatkan Objek), tambahkan parameter string kueri dan nilainya. Saat menandata ngani, Anda tidak melakukan encoding atas nilai-nilai ini; namun, saat membuat permintaan, Anda harus melakukan encoding atas nilai-nilai parameter ini. Parameter string kueri dalam permintaan DAPATKAN meliputi `response-content-type` , `response-content-language` , `response-expires` , `response-cache-control` , `response-content-disposition` , dan `response-content-encoding` . Parameter string `delete` kueri harus disertakan saat Anda membuat permintaan Delete CanonicalizedResource untuk multi-objek.

Elemen CanonicalizedResource yang berasal dari HTTP request-URI harus ditandatangani secara harfiah seperti yang muncul dalam permintaan HTTP, termasuk karakter meta pengkodean URL.

`CanonicalizedResource` mungkin berbeda dengan URI Permintaan HTTP. Khususnya, jika permintaan Anda menggunakan `Host` header HTTP untuk menetapkan bucket, maka bucket tersebut tidak akan muncul dalam URI Permintaan HTTP. Namun, `CanonicalizedResource` harus tetap menyertakan bucket. Parameter string kueri mungkin juga muncul dalam URI Permintaan tetapi tidak disertakan dalam `CanonicalizedResource`. Untuk informasi selengkapnya, lihat Bucket dengan hosting virtual.

Membuat CanonicalizedAmzHeaders elemen

Untuk membuat CanonicalizedAmzHeaders bagian dari`StringToSign`, pilih semua header permintaan HTTP yang dimulai dengan 'x-amz-' (menggunakan perbandingan case-insensitive), dan gunakan proses berikut.

CanonicalizedAmzHeaders proses

1	Ubah setiap nama header HTTP menjadi huruf kecil. Misalnya, 'X-Amz-Date ' menjadi 'x-amz-date '.
2	Urutkan pengumpulan header secara leksikografis berdasarkan nama header.
3	Gabungkan bidang header dengan nama yang sama menjadi satu pasangan “header-name:comma-separated-value-list” seperti yang ditentukan oleh RFC 2616, bagian 4.2, tanpa spasi di antara nilai. Misalnya, dua metadata header 'x-amz-meta-username: fred' dan 'x-amz-meta-username: barney ' akan digabungkan ke dalam header tunggal 'x-amz-meta-username: fred,barney '.
4	“Buka” header panjang yang mencakup beberapa baris (sebagaimana yang diizinkan oleh RFC 2616, bagian 4.2) dengan mengganti spasi terlipat (termasuk baris baru) dengan satu spasi.
5	Hapus setiap spasi di sekitar titik dua yang terdapat dalam header. Contohnya, header 'x-amz-meta-username: fred,barney ' akan menjadi 'x-amz-meta-username:fred,barney '
6	Terakhir, tambahkan karakter baris baru (U+000A) untuk setiap header kanonikalisasi dalam daftar hasil. Membuat elemen CanonicalizedResource dengan menggabungkan semua header dalam daftar ini menjadi satu string tunggal.

ELemen header posisi versus elemen StringToSign header HTTP yang diberi nama

Beberapa elemen header pertama StringToSign (Jenis-Konten, Tanggal, dan Konten-MD5) bersifat posisional. StringToSign tidak menyertakan nama-nama dari header ini, hanya nilai-nilainya saja dari permintaan. Sebaliknya, elemen 'x-amz-' diberi nama. Baik nama header dan nilai header akan muncul di StringToSign.

Jika header posisional yang disebutkan dalam definisi StringToSign tidak ada dalam permintaan Anda (misalnya, Content-Type atau Content-MD5 bersifat opsional untuk permintaan LETAKKAN dan tidak penting untuk permintaan DAPATKAN), ganti string kosong ("") untuk posisi tersebut.

Ketentuan stempel waktu

Stempel waktu yang valid (baik menggunakan header HTTP `Date` atau alternatif `x-amz-date`) wajib untuk permintaan yang diautentikasi. Selain itu, stempel waktu klien yang disertakan dengan permintaan yang diautentikasi harus dalam waktu 15 menit dari waktu sistem Amazon S3 saat permintaan tersebut diterima. Jika tidak, permintaan akan gagal dengan kode kesalahan `RequestTimeTooSkewed`. Tujuan dari pembatasan ini adalah untuk membatasi kemungkinan bahwa permintaan yang diintersepsi dapat diputar ulang oleh pihak lawan. Untuk perlindungan yang lebih kuat terhadap intervensi, gunakan transportasi HTTPS untuk permintaan yang diautentikasi.

> **Note**
>
> Batasan validasi pada tanggal permintaan hanya berlaku untuk permintaan yang diautentikasi yang tidak menggunakan autentikasi string kueri. Untuk informasi selengkapnya, lihat Alternatif autentikasi permintaan string kueri.

Beberapa pustaka klien HTTP tidak memaparkan kemampuan untuk mengatur header `Date` untuk permintaan. Jika Anda mengalami masalah saat menyertakan nilai header 'Tanggal' di header kanonikalisasi, Anda dapat mengatur stempel waktu untuk permintaan tersebut dengan menggunakan header '`x-amz-date`'. Nilai dari header `x-amz-date` harus dalam salah satu format RFC 2616 (http://www.ietf.org/rfc/rfc2616.txt). Saat header `x-amz-date` terdapat dalam permintaan, maka sistem akan mengabaikan header `Date` saat melakukan komputasi tanda tangan permintaan. Oleh karena itu, jika Anda menyertakan header `x-amz-date`, gunakan string kosong untuk `Date` saat membangun `StringToSign`. Lihat contohnya di bagian berikut.

Contoh-contoh autentikasi

Contoh-contoh dalam bagian ini menggunakan kredensial (nonkerja) dalam tabel berikut.

Parameter	Nilai
AWSAccessKeyId	`AKIAIOSFODNN7EXAMPLE`
AWSSecret AccessKey	`wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`

Dalam contoh `StringToSign`, format tidak signifikan, dan `\n` berarti titik kode Unicode `U+000A` yang biasanya disebut sebagai baris baru. Selain itu, contoh tersebut menggunakan “+0000” untuk menentukan zona waktu. Anda dapat menggunakan “GMT” untuk menentukan zona waktu, tetapi tanda tangan yang ditunjukkan dalam contoh tersebut akan berbeda.

Objek DAPATKAN

Contoh ini mendapatkan objek dari bucket awsexamplebucket1.

Permintaan	StringToSign

`GET /photos/puppy.jpg HTTP/1.1`
`Host: awsexamplebucket1.us-west-1.s3.amazonaws.com`
`Date: Tue, 27 Mar 2007 19:36:42 +0000`
 	
`Authorization: AWS AKIAIOSFODNN7EXAMPLE:qgk2+6Sv9/oM7G3qLEjTH1a1l1g=`
	`GET\n`
	`\n`
	`\n`
	`Tue, 27 Mar 2007 19:36:42 +0000\n`
	`/awsexamplebucket1/photos/puppy.jpg`

Perhatikan bahwa CanonicalizedResource menyertakan nama bucket, tetapi HTTP request-URI tidak. (Bucket ditentukan berdasarkan header Host.)

> **Note**
>
> Skrip Python berikut ini menghitung tanda tangan sebelumnya, menggunakan parameter yang disediakan. Anda dapat menggunakan skrip ini untuk membuat tanda tangan Anda sendiri, mengganti kunci dan StringToSign yang sesuai.

```
import base64
import hmac
from hashlib import sha1

access_key = 'AKIAIOSFODNN7EXAMPLE'.encode("UTF-8")
secret_key = 'wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY'.encode("UTF-8")

string_to_sign = 'GET\n\n\nTue, 27 Mar 2007 19:36:42 +0000\n/awsexamplebucket1/
photos/puppy.jpg'.encode("UTF-8")
signature = base64.b64encode(
```

```
                                hmac.new(
                                         secret_key, string_to_sign, sha1
                                         ).digest()
                                ).strip()


print(f"AWS {access_key.decode()}:{signature.decode()}")
```

Objek LETAKKAN

Contoh ini menempatkan sebuah objek ke dalam bucket awsexamplecket1.

Permintaan	StringToSign

`PUT /photos/puppy.jpg HTTP/1.1`
`Content-Type: image/jpeg`
`Content-Length: 94328`
`Host: awsexamplebucket1.s3.us-wes`
`t-1.amazonaws.com`
`Date: Tue, 27 Mar 2007 21:15:45 +0000`
 	
`Authorization: AWS AKIAIOSFODNN7EXAMP`
`LE:`
`iqRzw+ileNPu1fhspnRs8nOjjIA=`
	`PUT\n`
	`\n`
	`image/jpeg\n`
	`Tue, 27 Mar 2007 21:15:45 +0000\n`
	`/awsexamplebucket1/photos/puppy.jpg`

Perhatikan header Content-Type dalam permintaan dan di. StringToSign Perhatikan juga bahwa Content-MD5 dibiarkan kosong di StringToSign, karena tidak ada dalam permintaan.

Daftar

Contoh ini mencantumkan konten dari bucket awsexamplebucket1.

Permintaan	StringToSign

`GET /?prefix=photos&max-keys=50&marker=puppy`
` HTTP/1.1`
`User-Agent: Mozilla/5.0`
	`GET\n`
	`\n`
	`\n`

Permintaan	StringToSign

`Host: awsexamplebucket1.s3.us-west-1.amazonaws.com`
`Date: Tue, 27 Mar 2007 19:42:41 +0000`
 	
`Authorization: AWS AKIAIOSFODNN7EXAMPLE:m0WP8eCtspQl5Ahe6L1SozdX9YA=`
	`Tue, 27 Mar 2007 19:42:41 +0000\n`
	`/awsexamplebucket1/`

Perhatikan garis miring pada CanonicalizedResource dan tidak adanya parameter string kueri.

Mengambil

Contoh ini mengambil sub-sumber daya kebijakan kontrol akses untuk bucket 'awsexamplebucket1'.

Permintaan	StringToSign

`GET /?acl HTTP/1.1`
`Host: awsexamplebucket1.s3.us-west-1.amazonaws.com`
`Date: Tue, 27 Mar 2007 19:44:46 +0000`
 	
`Authorization: AWS AKIAIOSFODNN7EXAMPLE:82ZHiFIjc+WbcwFKGUVEQspPn+0=`
	`GET\n`
	`\n`
	`\n`
	`Tue, 27 Mar 2007 19:44:46 +0000\n`
	`/awsexamplebucket1/?acl`

Perhatikan bagaimana parameter string kueri sub-sumber daya disertakan di CanonicalizedResource.

Hapus

Contoh ini menghapus objek dari bucket 'awsexamplebucket1' menggunakan alternatif gaya penulisan path dan Tanggal.

Permintaan	StringToSign

`DELETE /awsexamplebucket1/photos/puppy.jpg HTTP/1.1`
`User-Agent: dotnet`
	`DELETE\n`
	`\n`
	`\n`

Permintaan	StringToSign

`Host: s3.us-west-1.amazonaws.com`
`Date: Tue, 27 Mar 2007 21:20:27 +0000`
 	
`x-amz-date: Tue, 27 Mar 2007 21:20:26`
` +0000`
`Authorization: AWS AKIAIOSFODNN7EXAMP`
`LE:XbyTlbQdu9Xw5o8P4iMwPktxQd8=`
	`Tue, 27 Mar 2007 21:20:26 +0000\n`
	`/awsexamplebucket1/photos/puppy.jpg`

Perhatikan bagaimana kami menggunakan metode 'x-amz-date' alternatif untuk menentukan tanggal (karena pustaka klien kami mencegah kami mengatur tanggal, katakanlah). Dalam hal ini, `x-amz-date` lebih diutamakan dari header `Date`. Oleh karena itu, entri tanggal pada tanda tangan harus berisi nilai header `x-amz-date`.

Unggah

Contoh ini mengunggah objek ke bucket yang dihosting virtual bergaya CNAME dengan metadata.

Permintaan	StringToSign

`PUT /db-backup.dat.gz HTTP/1.1`
`User-Agent: curl/7.15.5`
`Host: static.example.com:8080`
`Date: Tue, 27 Mar 2007 21:06:08 +0000`
 	
`x-amz-acl: public-read`
`content-type: application/x-download`
`Content-MD5: 4gJE4saaMU4BqNR0kLY+lw==`
`X-Amz-Meta-ReviewedBy: joe@example.com`
`X-Amz-Meta-ReviewedBy: jane@exam`
`ple.com`
`X-Amz-Meta-FileChecksum: 0x02661779`
`X-Amz-Meta-ChecksumAlgorithm: crc32`
`Content-Disposition: attachment;`
` filename=database.dat`
`Content-Encoding: gzip`
`Content-Length: 5913339`
 	
` Authorization: AWS AKIAIOSFODNN7EXAMP`
`LE:`
	`PUT\n`
	`4gJE4saaMU4BqNR0kLY+lw==\n`
	`application/x-download\n`
	`Tue, 27 Mar 2007 21:06:08 +0000\n`
	
	`x-amz-acl:public-read\n`
	`x-amz-meta-checksumalgorithm:c`
	`rc32\n`
	`x-amz-meta-filechecksum:0x026`
	`61779\n`
	`x-amz-meta-reviewedby:`
	`joe@example.com,jane@example.com`
	`\n`
	`/static.example.com/db-backup.dat`
	`.gz`

Permintaan	StringToSign
`jtBQa0Aq+DkULFI8qrpwIjGEx0E=`	

Perhatikan bagaimana header 'x-amz-' diurutkan, dipotong spasi ekstranya, dan diubah menjadi huruf kecil. Perhatikan juga bahwa beberapa header yang memiliki nama yang sama telah digabungkan menggunakan koma untuk memisahkan nilai.

Perhatikan bagaimana hanya header entitas HTTP `Content-Type` dan `Content-MD5` yang muncul di `StringToSign`. Header entitas `Content-*` lainnya tidak muncul.

Sekali lagi, perhatikan bahwa `CanonicalizedResource` menyertakan nama bucket, tetapi URI Permintaan HTTP tidak menyertakannya. (Bucket ditentukan berdasarkan header Host.)

Tuliskan semua bucket saya

Permintaan	StringToSign

`GET / HTTP/1.1`
`Host: s3.us-west-1.amazonaws.com`
`Date: Wed, 28 Mar 2007 01:29:59 +0000`
 	
`Authorization: AWS AKIAIOSFODNN7EXAMPLE:qGdzdERIC03wnaRNKh6OqZehG9s=`
	`GET\n`
	`\n`
	`\n`
	`Wed, 28 Mar 2007 01:29:59 +0000\n`
	`/`

Kunci Unicode

Permintaan	StringToSign

`GET /dictionary/fran%C3%A7ais/pr%c3%a9f%c3%a8re HTTP/1.1`
`Host: s3.us-west-1.amazonaws.com`
`Date: Wed, 28 Mar 2007 01:49:49 +0000`
`Authorization: AWS AKIAIOSFODNN7EXAMPLE:DNEZGsoieTZ92F3bUfSPQcbGmlM=`
	`GET\n`
	`\n`
	`\n`
	`Wed, 28 Mar 2007 01:49:49 +0000\n`
	`/dictionary/fran%C3%A7ais/pr%c3%a9f%c3%a8re`

> **Note**
>
> Elemen dalam `StringToSign` yang berasal dari URI Permintaan benar-benar diambil, termasuk URL-Encoding dan kapitalisasi.

Masalah penandatanganan permintaan REST

Saat autentikasi permintaan REST gagal, sistem akan merespons permintaan tersebut dengan dokumen kesalahan XML. Informasi yang terdapat dalam dokumen kesalahan ini ditujukan untuk membantu pengembang melakukan diagnosis masalah. Secara khusus, elemen `StringToSign` dari dokumen kesalahan `SignatureDoesNotMatch` memberi tahu Anda dengan tepat kanonikalisasi permintaan yang digunakan oleh sistem.

Beberapa toolkit diam-diam memasukkan header yang tidak Anda ketahui sebelumnya, misalnya menambahkan header `Content-Type` selama LETAKKAN. Dalam sebagian besar kasus ini, nilai header yang dimasukkan tetap konstan, sehingga Anda dapat menemukan header yang hilang dengan menggunakan alat seperti Ethereal atau tcpmon.

Alternatif autentikasi permintaan string kueri

Anda dapat melakukan autentikasi atas jenis permintaan tertentu dengan meneruskan informasi yang diperlukan sebagai parameter string kueri dan bukan dengan menggunakan header HTTP `Authorization`. Ini berguna untuk mengaktifkan akses peramban pihak ketiga langsung ke data Amazon S3 pribadi Anda tanpa mengajukan permintaan melalui proxy. Tujuannya adalah untuk membuat permintaan yang "sudah ditandatangani sebelumnya" dan mengkodekannya sebagai URL yang dapat diambil oleh peramban pengguna akhir. Selain itu, Anda dapat membatasi sebuah permintaan yang telah ditandatangani sebelumnya dengan menentukan waktu kedaluwarsa.

Untuk informasi selengkapnya mengenai penggunaan parameter kueri untuk melakukan autentikasi permintaan, lihat Autentikasi Permintaan: Menggunakan Parameter Kueri (Tanda Tangan Versi 4 AWS) di Referensi API Amazon Simple Storage Service. Sebagai contoh penggunaan SDK AWS untuk membuat URL yang telah ditandatangani sebelumnya, lihat Berbagi objek dengan URL yang telah ditandatangani.

Membuat tanda tangan

Berikut ini adalah contoh permintaan REST Amazon S3 string kueri yang diautentikasi.

```
GET /photos/puppy.jpg
?AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&Expires=1141889120&Signature=vjbyPxybdZaNmGa
%2ByT272YEAiv4%3D HTTP/1.1
Host: awsexamplebucket1.s3.us-west-1.amazonaws.com
Date: Mon, 26 Mar 2007 19:37:58 +0000
```

Metode autentikasi permintaan string kueri tidak memerlukan header HTTP khusus. Sebaliknya, elemen autentikasi yang diperlukan akan ditentukan sebagai parameter string kueri:

Nama parameter string kueri	Nilai contoh	Deskripsi
`AWSAccessKeyId`	`AKIAIOSFODNN7EXAMPLE`	Kunci akses ID AWS Anda. Menentukan kunci akses rahasia AWS yang digunakan untuk menandatangani permintaan dan secara tidak langsung, menentukan identitas pengembang yang mengajukan permintaan tersebut.
`Expires`	`1141889120`	Waktu berakhirnya tanda tangan, ditetapkan sebagai jumlah detik sejak epoch (00:00:00 UTC pada tanggal 1 Januari 1970). Permintaan yang diterima setelah waktu ini (berdasarkan waktu server) akan ditolak.
`Signature`	`vjbyPxybdZaNmGa%2ByT272YEAiv4%3D`	Encoding URL untuk encoding Base64 HMAC-SHA1 dari StringToSign.

Metode autentikasi permintaan string kueri sedikit berbeda dari metode biasa tetapi hanya dalam bentuk format dari parameter permintaan `Signature` dan elemen `StringToSign`. Berikut ini adalah pseudo-grammar yang mengilustrasikan metode autentikasi permintaan string kueri.

```
Signature = URL-Encode( Base64( HMAC-SHA1( YourSecretAccessKey, UTF-8-Encoding-
Of( StringToSign ) ) ) );
```

```
StringToSign = HTTP-VERB + "\n" +
    Content-MD5 + "\n" +
    Content-Type + "\n" +
    Expires + "\n" +
    CanonicalizedAmzHeaders +
    CanonicalizedResource;
```

`YourSecretAccessKey` adalah kunci akses rahasia ID AWS yang diberikan Amazon kepada Anda saat mendaftar menjadi pengembang Amazon Web Service. Perhatikan bagaimana `Signature` URL dikodekan agar sesuai untuk penempatan dalam string kueri. Perhatikan juga bahwa dalam `StringToSign`, elemen posisi `Date` HTTP telah diganti dengan `Expires`. `CanonicalizedAmzHeaders` dan `CanonicalizedResource` sama.

> **Note**
>
> Dalam metode autentikasi string kueri, Anda tidak menggunakan `Date` atau `x-amz-date` `request` saat menghitung string yang akan ditandatangani.

Autentikasi permintaan string kueri

Permintaan	StringToSign

`GET /photos/puppy.jpg?AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE&`
`  Signature=NpgCjnDzrM%2BWFzoENXmpNDUsSn8%3D&`
`  Expires=1175139620 HTTP/1.1`
 	
`Host: awsexamplebucket1.s3.us-west-1.amazonaws.com`
	`GET\n`
	`\n`
	`\n`
	`1175139620\n`
	
	`/awsexamplebucket1/photos/puppy.jpg`

Kami menganggap bahwa ketika peramban membuat permintaan DAPATKAN, peramban tidak akan menyediakan header Konten-MD5 atau header Jenis Konten, dan tidak akan mengatur header x-amz-, sehingga bagian-bagian dari `StringToSign` dibiarkan kosong.

Menggunakan pengodean Base64

Tanda tangan permintaan HMAC harus dikodekan dengan Base64. Pengodean Base64 mengubah tanda tangan menjadi string ASCII sederhana yang dapat dilampirkan ke permintaan. Karakter yang dapat muncul dalam string tanda tangan seperti tambah (+), garis miring (/), dan sama dengan (=) harus dikodekan jika digunakan dalam URI. Misalnya, jika kode autentikasi menyertakan tanda tambah (+), konversikan kode tersebut sebagai %2B dalam permintaan. Konversikan kode garis miring menjadi %2F dan sama dengan menjadi %3D.

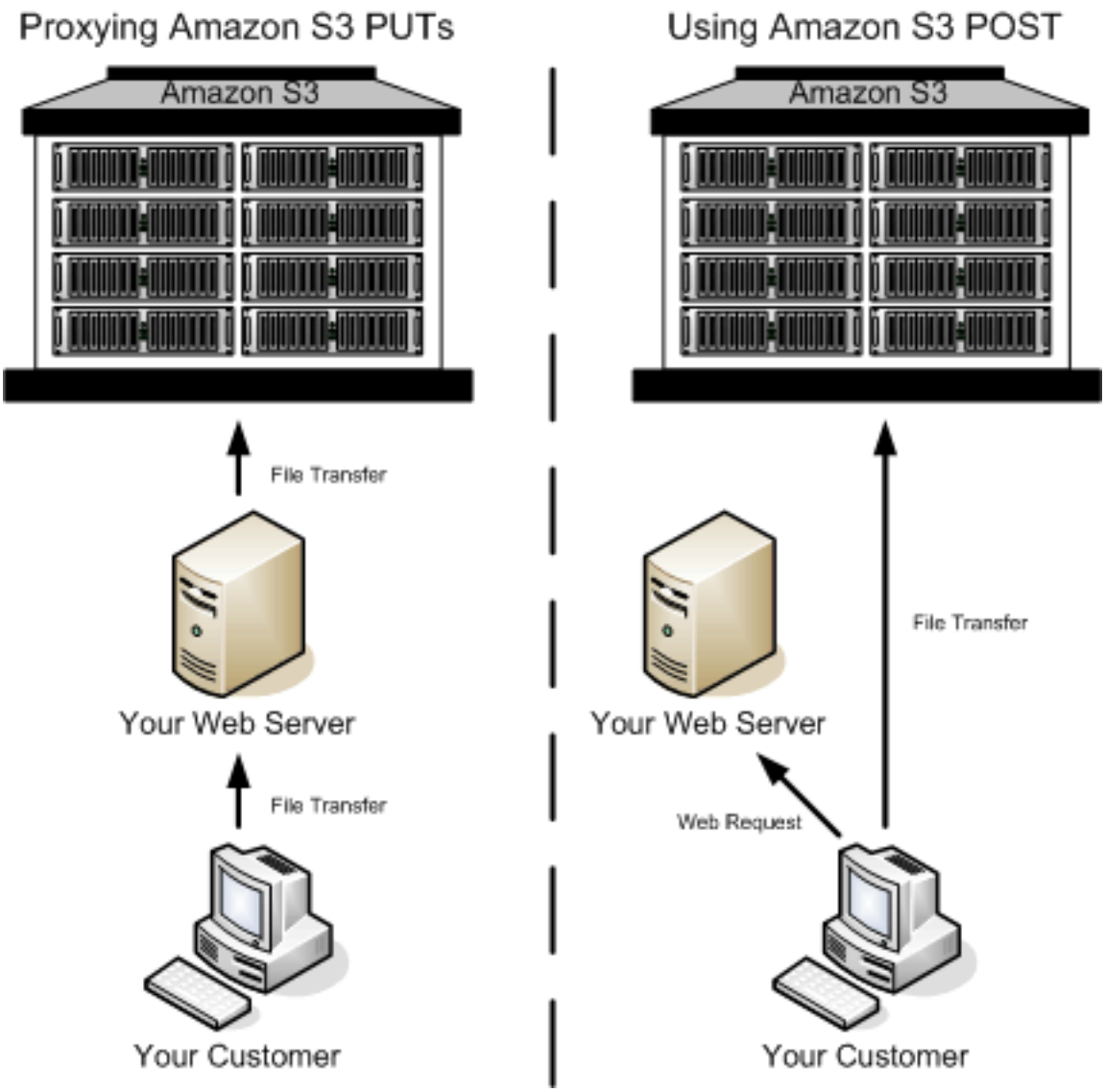
Untuk contoh pengodean Base64, lihat Amazon S3 Contoh-contoh autentikasi.

Unggahan berbasis browser menggunakan POST (versi AWS tanda tangan 2)

Amazon S3 mendukung POST, yang memungkinkan pengguna mengunggah konten secara langsung ke Amazon S3. POST dirancang untuk menyederhanakan unggahan, mengurangi latensi unggahan, dan menghemat biaya Anda pada aplikasi tempat pengguna mengunggah data untuk disimpan di Amazon S3.

> **Note**
>
> Otentikasi permintaan yang dibahas di bagian ini didasarkan pada AWS Signature Version 2, protokol untuk mengautentikasi permintaan API masuk ke layanan. AWS Amazon S3 sekarang mendukung Signature Version 4, protokol untuk mengautentikasi permintaan API masuk ke AWS layanan, secara keseluruhan. Wilayah AWS Pada saat ini, Wilayah AWS dibuat sebelum 30 Januari 2014 akan terus mendukung protokol sebelumnya, Signature Version 2. Wilayah baru setelah tanggal 30 Januari 2014 hanya akan mendukung Tanda Tangan Versi 4 dan oleh karena itu semua permintaan ke wilayah tersebut harus dilakukan dengan Tanda Tangan Versi 4. Untuk informasi selengkapnya, lihat Mengautentikasi Permintaan di Unggahan Berbasis Browser Menggunakan POST (Versi AWS Tanda Tangan 4) di Referensi API Layanan Penyimpanan Sederhana Amazon.

Gambar berikut menunjukkan unggahan menggunakan Amazon S3 POST.



Mengunggah menggunakan POST

1	Pengguna membuka peramban web dan mengakses halaman web Anda.
2	Halaman web Anda berisi formulir HTTP yang memuat semua informasi yang diperlukan pengguna untuk mengunggah konten ke Amazon S3.
3	Pengguna mengunggah konten secara langsung ke Amazon S3.

> **Note**
>
> Autentikasi string kueri tidak didukung untuk POST.

Formulir HTML (versi AWS tanda tangan 2)

Topik

- Pengodean formulir HTML
- Deklarasi formulir HTML
- Bidang formulir HTML
- Konstruksi kebijakan
- Membuat tanda tangan
- Pengalihan

Saat berkomunikasi dengan Amazon S3, Anda biasanya menggunakan REST atau SOAP API untuk menjalankan, mendapatkan, menghapus, dan melakukan operasi lainnya. Dengan POST, pengguna mengunggah data secara langsung ke Amazon S3 melalui peramban mereka, yang tidak dapat memproses SOAP API atau membuat permintaan REST PUT.

> **Note**
>
> Dukungan SOAP melalui HTTP tidak digunakan lagi, tetapi SOAP masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Alih-alih menggunakan SOAP, kami sarankan Anda menggunakan REST API atau AWS SDK.

Agar pengguna dapat mengunggah konten ke Amazon S3 dengan menggunakan peramban mereka, Anda perlu menggunakan formulir HTML. Formulir HTML terdiri dari deklarasi formulir dan bidang formulir. Deklarasi formulir berisi informasi tingkat tinggi mengenai permintaan tersebut. Kolom formulir berisi informasi terperinci tentang permintaan, serta kebijakan yang digunakan untuk mengautentikasi permintaan tersebut dan memastikan bahwa permintaan tersebut memenuhi persyaratan yang Anda tentukan.

> **Note**
>
> Data formulir dan batasannya (tidak termasuk isi file) tidak boleh lebih dari 20 KB.

Bagian ini menjelaskan cara menggunakan formulir HTML.

Pengodean formulir HTML

Formulir dan kebijakan harus menggunakan pengodean UTF-8. Anda dapat menerapkan pengodean UTF-8 pada formulir tersebut dengan menetapkannya dalam judul HTML-nya atau sebagai header permintaan.

> **Note**
>
> Deklarasi formulir HTML tidak menerima parameter-parameter autentikasi string kueri.

Berikut ini adalah contoh pengodean UTF-8 pada judul HTML:

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    ...
  </head>
  <body>
```

Berikut ini adalah contoh pengodean UTF-8 pada header permintaan:

```
Content-Type: text/html; charset=UTF-8
```

Deklarasi formulir HTML

Deklarasi formulir memiliki tiga komponen, yakni: aksi, metode, dan jenis enklosure. Jika salah satu dari nilai ini tidak sesuai, maka permintaan tersebut gagal.

Tindakan ini menentukan URL yang memproses permintaan, yang harus diatur ke URL bucket. Sebagai contoh, jika nama bucket Anda adalah `awsexamplebucket1` dan Wilayahnya adalah AS Barat (California Utara), URL-nya adalah `https://awsexamplebucket1.s3.us-west-1.amazonaws.com/`.

> **Note**
>
> Nama kunci ditentukan dalam bidang formulir.

Metodenya harus POST.

Jenis enklosur (enctype) harus ditentukan dan harus diatur untuk data-banyak-bagian/formulir baik untuk pengunggahan file maupun pengunggahan bidang teks. Untuk informasi selengkapnya, buka RFC 1867.

Example

Contoh berikut adalah deklarasi formulir untuk bucket “awsexamplebucket1”.

```
<form action="https://awsexamplebucket1.s3.us-west-1.amazonaws.com/" method="post"

enctype="multipart/form-data">
```

Bidang formulir HTML

Tabel berikut menjelaskan bidang yang dapat digunakan dalam sebuah formulir HTML.

> **Note**
>
> Variabel `${filename}` secara otomatis diganti dengan nama file yang diberikan oleh pengguna dan dikenali oleh semua bidang formulir. Jika peramban atau klien menyediakan jalur penuh atau sebagian ke file, maka hanya teks yang ada setelah garis miring terakhir (/) atau garsi miring terbalik (\) yang akan digunakan. Sebagai contoh, "C:\Program Files \directory1\file.txt" akan diinterpretasikan sebagai "file.txt". Jika tidak ada nama file atau file yang diberikan, maka variabel diganti dengan string kosong.

Nama bidang	Deskripsi	Diperlukan
`AWSAccessKeyId`	ID Kunci AWS Akses pemilik bucket yang memberikan akses kepada pengguna anonim untuk permintaan yang memenuhi serangkai	Bersyarat

Nama bidang	Deskripsi	Diperlukan
	an batasan dalam kebijakan. Kolom ini wajib diisi jika permintaan menyertakan dokumen kebijakan.	
`acl`	Daftar kontrol akses (ACL) Amazon S3. Jika ternyata daftar kontrol akses yang ditentuka n tidak valid, maka akan muncul pesan kesalahan. Untuk informasi selengkapnya tentang ACL, lihat Daftar kontrol akses (ACL). Jenis: String Default: privat Nilai Valid: `private | public-read | public-read-write | aws-exec-read | authenticated-read | bucket-owner-read | bucket-ow ner-full-control`	Tidak
`Cache-Control, Content-Type, Content-Disposition, Conten t-Encoding, Expires`	Header khusus REST. Untuk informasi lebih lanjut, lihat PUT Objek.	Tidak

Nama bidang	Deskripsi	Diperlukan
`key`	Nama kunci yang diunggah. Untuk menggunakan nama file yang diberikan oleh pengguna, gunakan variabel ${filename}. Contohnya, jika seorang pengguna benama Betty mengunggah file lolcatz.jpg dan Anda menentukan /user/betty/${filename}, file tersebut disimpan sebagai /user/betty/lolcatz.jpg. Untuk informasi selengkapnya, lihat Bekerja dengan metadata objek.	Ya
`policy`	Kebijakan keamanan yang menjelaskan apa yang diizinkan dalam permintaan. Permintaan yang tidak memiliki kebijakan keamanan dianggap anonim dan hanya akan berhasil pada bucket yang dapat ditulis oleh publik.	Tidak

Nama bidang	Deskripsi	Diperlukan
`success_action_redirect, redirect`	URL yang akan diarahkan klien setelah unggahan berhasil. Amazon S3 menambahkan nilai bucket, kunci, dan etag sebagai parameter string kueri ke URL. Jika success_action_redirect tidak ditentukan, maka Amazon S3 akan mengembalikan jenis dokumen kosong yang ditentukan dalam bidang success_action_status. Jika Amazon S3 tidak dapat menginterpretasikan URL, maka Amazon S3 akan mengabaikan bidang tersebut. Jika pengunggahan gagal, maka Amazon S3 akan menampilkan pesan kesalahan dan tidak akan mengalihkan pengguna ke URL. Untuk informasi selengkapnya, lihat Pengalihan. **Note** Nama bidang pengalihan telah dihilangkan dan dukungan untuk nama bidang pengalihan akan dihapus di masa mendatang.	Tidak

Nama bidang	Deskripsi	Diperlukan
`success_action_status`	Kode status dikembalikan kepada klien setelah berhasil mengunggah jika success_action_redirect tidak ditentukan. Nilai yang valid adalah 200, 201, atau 204 (default). Jika nilai diatur ke 200 atau 204, maka Amazon S3 akan mengembalikan dokumen kosong dengan kode status 200 atau 204. Jika nilai diatur ke 201, maka Amazon S3 akan mengembalikan dokumen XML dengan kode status 201. Untuk informasi tentang konten dokumen XML, lihat Objek POST. Jika nilai tidak diatur atau jika diatur dengan nilai yang tidak valid, maka Amazon S3 akan mengembalikan dokumen kosong dengan kode status 204. **Note** Beberapa versi dari Adobe Flash player tidak menangani respons HTTP dengan baik pada bodi kosong. Untuk mendukung pengunggahan melalui Adobe Flash, sebaiknya atur `success_action_status` menjadi 201.	Tidak

Nama bidang	Deskripsi	Diperlukan
`signature`	Tanda tangan HMAC dibangun dengan menggunakan kunci akses rahasia yang sesuai dengan yang disediakanAWSAccessKeyId. Bidang ini wajib diisi jika dokumen kebijakan disertakan bersama permintaan. Untuk informasi selengkapnya, lihat Identity and Access Management untuk Amazon S3.	Bersyarat
`x-amz-security-token`	Token keamanan digunakan oleh kredensial sesi Jika permintaan menggunakan Amazon DevPay maka diperlukan dua bidang `x-amz-security-token` formulir: satu untuk token produk dan satu untuk token pengguna. Jika permintaan menggunakan kredensial sesi, maka permintaan tersebut memerlukan satu formulir `x-amz-security-token` . Untuk informasi selengkapnya, lihat Kredensial Keamanan Sementara dalam Panduan Pengguna IAM.	Tidak
Nama bidang lain diawali dengan x-amz-meta -	Metadata yang ditentukan pengguna. Amazon S3 tidak memvalidasi atau menggunakan data ini. Untuk informasi selengkapnya, lihat PUT Objek.	Tidak

Nama bidang	Deskripsi	Diperlukan
file	Konten file atau teks. File atau konten harus menjadi bidang terakhir dalam formulir. Bidang apa pun di bawahnya akan diabaikan. Anda tidak dapat mengunggah lebih dari satu file sekaligus.	Ya

Konstruksi kebijakan

Topik

- Kedaluwarsa
- Ketentuan
- Pencocokan ketentuan
- Pelarian karakter

Kebijakan ini adalah dokumen JSON dengan pengodean UTF-8 dan pengodean Base64 yang menetapkan ketentuan-ketentuan yang harus dipenuhi permintaan dan digunakan untuk melakukan autentikasi konten. Tergantung pada cara Anda merancang dokumen kebijakan, Anda dapat menggunakannya untuk setiap unggahan, setiap pengguna, untuk semua unggahan, atau sesuai dengan desain lain yang memenuhi kebutuhan Anda.

> **Note**
>
> Meskipun dokumen kebijakan bersifat opsional, seabiknya buat bucket yang dapat ditulis secara publik.

Berikut ini adalah contoh dari dokumen kebijakan:

```
{ "expiration": "2007-12-01T12:00:00.000Z",

  "conditions": [
```

```
    {"acl": "public-read" },

    {"bucket": "awsexamplebucket1" },

    ["starts-with", "$key", "user/eric/"],

  ]

}
```

Dokumen kebijakan berisi masa kedaluwarsa dan ketentuan-ketentuan.

Kedaluwarsa

Elemen kedaluwarsa menetapkan tanggal kedaluwarsa kebijakan dengan format tanggal ISO 8601 UTC. Contohnya, “2007-12-01T12:00:00.000Z” menyatakan bahwa kebijakan tersebut tidak berlaku setelah UTC tengah malam pada tanggal 1 Desember 2007. Kedaluwarsa wajib ada dalam kebijakan.

Ketentuan

Ketentuan dalam dokumen kebijakan memvalidasi konten dari objek yang diunggah. Setiap kolom formulir yang Anda tentukan dalam formulir (kecualiAWSAccessKeyId, tanda tangan, file, kebijakan, dan nama bidang yang memiliki awalan x-ignore-) harus disertakan dalam daftar kondisi.

> **Note**
>
> Jika Anda memiliki beberapa bidang dengan nama yang sama, maka nilai-nilainya harus dipisahkan dengan koma. Misalnya, jika Anda memiliki dua bidang bernama "x-amz-meta-tag" dan yang pertama memiliki nilai “Ninja” dan yang kedua memiliki nilai “Stallman”, Anda akan menetapkan dokumen kebijakan ke. `Ninja,Stallman`
> Semua variabel yang ada dalam formulir diperluas sebelum kebijakan tersebut divalidasi. Oleh karena itu, semua pencocokan kondisi harus dilakukan pada kolom-kolom yang diperluas. Misalnya, jika Anda mengatur bidang kunci ke `user/betty/${filename}`, kebijakan Anda mungkin `[ "starts-with", "$key", "user/betty/" ]`. Jangan masukkan `[ "starts-with", "$key", "user/betty/${filename}" ]`. Untuk informasi selengkapnya, lihat Pencocokan ketentuan.

Tabel berikut menjelaskan ketentuan dokumen kebijakan.

Nama elemen	Deskripsi
acl	Menentukan ketentuan yang harus dipenuhi oleh ACL. Dukung pencocokan yang tepat dan `starts-with` .
content-length-range	Menentukan ukuran minimal dan maksimal yang diizinkan untuk konten yang diunggah. Dukung penyesuaian rentang.
Kontrol Cache, Jenis Konten, Disposisi Konten, Pengodean Konten, Kedaluwarsa	Header khusus REST. Dukung pencocokan yang tepat dan `starts-with` .
kunci	Nama kunci yang diunggah. Dukung pencocokan yang tepat dan `starts-with` .
success_action_redirect, mengalihkan	URL yang akan diarahkan klien setelah unggahan berhasil. Dukung pencocokan yang tepat dan `starts-with` .
success_action_status	Kode status dikembalikan kepada klien setelah berhasil mengunggah jika success_action_redirect tidak ditentukan. Dukung pencocokan yang tepat.
x-amz-security-token	Token DevPay keamanan Amazon. Setiap permintaan yang menggunakan Amazon DevPay memerlukan dua bidang `x-amz-security-token` formulir: satu untuk token produk dan satu untuk token pengguna. Akibatnya, nilai harus dipisahkan dengan koma.

Nama elemen	Deskripsi
	Contoh, jika token pengguna adalah `eW91dHViZQ==` dan token produk adalah `b0hnNVNKWVJIQTA=` , maka Anda harus mengatur entri kebijakan menjadi: `{ "x-amz-security-token": "eW91dHViZQ==,b0hnNVNKWVJIQTA=" }` .
Nama bidang lain diawali dengan x-amz-meta -	Metadata yang ditentukan pengguna. Dukung pencocokan yang tepat dan `starts-with` .

> **Note**
>
> Jika alat memberikan bidang tambahan (misalnya, Flash menambahkan nama file), maka Anda harus menambahkannya pada dokumen kebijakan. Jika Anda dapat mengontrol fungsi ini, tambahkan awalan `x-ignore-` pada bidang tersebut sehingga Amazon S3 mengabaikan fitur tersebut dan tidak akan memengaruhi versi fitur ini di masa mendatang.

Pencocokan ketentuan

Tabel berikut menjelaskan jenis pencocokan ketentuan. Meskipun Anda harus menentukan satu ketentuan untuk setiap bidang formulir yang disebutkan dalam formulir tersebut, Anda dapat membuat kriteria pencocokan yang lebih kompleks dengan menetapkan beberapa ketentuan untuk bidang formulir.

Ketentuan	Deskripsi
Kecocokan yang Tepat	Kecocokan yang tepat memverifikasi bahwa kolom cocok dengan nilai spesifik. Contoh ini menunjukkan bahwa ACL harus diatur ke baca publik: `{"acl": "public-read" }` Contoh ini adalah cara alternatif untuk menunjukkan bahwa ACL harus diatur ke baca publik:

Ketentuan	Deskripsi
	`[ "eq", "$acl", "public-read" ]`
Dimulai Dengan	Jika nilai harus dimulai dengan nilai tertentu, gunakan mulai-dengan. Contoh ini menunjukkan bahwa kunci harus dimulai dengan pengguna/betty: `["starts-with", "$key", "user/betty/"]`
Mencocokkan Konten Apa Pun	Untuk mengonfigurasi kebijakan yang mengizinkan konten apa pun di dalam sebuah bidang, gunakan mulai-dengan dengan nilai kosong. Contoh ini memungkinkan success_action_redirect: `["starts-with", "$success_action_redirect", ""]`
Menentukan Rentang	Untuk kolom yang menerima rentang, pisahkan rentang atas dan bawah dengan koma. Contoh ini memungkinkan ukuran file dari 1 hingga 10 megabyte: `["content-length-range", 1048579, 10485760]`

Pelarian karakter

Tabel berikut menjelaskan pelarian karakter dari dalam dokumen kebijakan.

Karakter pelarian	Deskripsi
\\	Garis miring terbalik
\$	Tanda dolar

Karakter pelarian	Deskripsi
\b	Spasi Mundur
\f	Umpan formulir
\n	Baris baru
\r	Kembali ke awal baris
\t	Tab Horizontal
\v	Tab vertikal
\u*xxxx*	Semua karakter Unicode

Membuat tanda tangan

Langkah	Deskripsi
1	Lakukan pengodean kebijakan dengan menggunakan UTF-8.
2	Lakukan pengodean UTF-8 byte tersebut dengan menggunakan Base64.
3	Tanda tangani kebijakan dengan kunci akses rahasia Anda dengan menggunakan HMAC SHA-1.
4	Lakukan pengodean pada tanda tangan SHA-1 dengan menggunakan Base64.

Untuk informasi umum tentang autentikasi, lihat Identity and Access Management untuk Amazon S3.

Pengalihan

Bagian ini menjelaskan cara menangani pengalihan.

Pengalihan umum

Setelah menyelesaikan permintaan POST, pengguna akan dialihkan ke lokasi yang telah Anda tentukan dalam bidang `success_action_redirect`. Jika Amazon S3 tidak dapat menginterpretasikan URL, maka Amazon S3 mengabaikan bidang `success_action_redirect`.

Jika `success_action_redirect` tidak ditentukan, maka Amazon S3 akan mengembalikan jenis dokumen kosong yang ditentukan dalam bidang `success_action_status`.

Jika permintaan POST gagal, maka Amazon S3 akan menampilkan kesalahan dan tidak akan memberikan pengalihan.

Pengalihan pra-unggahan

Jika bucket Anda dibuat menggunakan < CreateBucketConfiguration >, pengguna akhir Anda mungkin memerlukan pengalihan. Jika hal ini terjadi, beberapa peramban mungkin menangani pengalihan dengan tidak benar. Hal ini tergolong jarang terjadi, namun kemungkinan besar akan terjadi setelah pembuatan bucket.

Unggah contoh (versi AWS tanda tangan 2)

Topik

- Unggahan file
- Unggahan bidang teks

> **Note**
>
> Otentikasi permintaan yang dibahas di bagian ini didasarkan pada AWS Signature Version 2, protokol untuk mengautentikasi permintaan API masuk ke layanan. AWS Amazon S3 sekarang mendukung Signature Version 4, protokol untuk mengautentikasi permintaan API masuk ke AWS layanan, secara keseluruhan. Wilayah AWS Pada saat ini, Wilayah AWS dibuat sebelum 30 Januari 2014 akan terus mendukung protokol sebelumnya, Signature Version 2. Wilayah baru setelah tanggal 30 Januari 2014 hanya akan mendukung Tanda Tangan Versi 4 dan oleh karena itu semua permintaan ke wilayah tersebut harus dilakukan dengan Tanda Tangan Versi 4. Untuk informasi selengkapnya, lihat Contoh:

Unggahan Berbasis Browser menggunakan HTTP POST (Menggunakan AWS Tanda Tangan Versi 4) di Referensi API Amazon Simple Storage Service.

Unggahan file

Contoh ini menunjukkan proses lengkap untuk membuat kebijakan dan formulir yang dapat digunakan untuk mengunggah lampiran file.

Pembuatan kebijakan dan formulir

Kebijakan berikut mendukung unggahan ke Amazon S3 untuk bucket awsexamplebucket1.

```
{ "expiration": "2007-12-01T12:00:00.000Z",
  "conditions": [
    {"bucket": "awsexamplebucket1"},
    ["starts-with", "$key", "user/eric/"],
    {"acl": "public-read"},
    {"success_action_redirect": "https://awsexamplebucket1.s3.us-west-1.amazonaws.com/
successful_upload.html"},
    ["starts-with", "$Content-Type", "image/"],
    {"x-amz-meta-uuid": "14365123651274"},
    ["starts-with", "$x-amz-meta-tag", ""]
  ]
}
```

Kebijakan ini mengharuskan:

- Unggahan harus dilakukan sebelum pukul 12:00 UTC tanggal 1 Desember 2007.
- Konten harus diunggah ke bucket awsexamplebucket1.
- Kunci harus dimulai dengan "user/eric/".
- ACL diatur menjadi baca publik.
- Success_action_redirect diatur ke https://awsexamplebucket1.s3.us-west-1.amazonaws.com/successful_upload.html.
- Objek adalah file gambar.
- x-amz-meta-uuid Tag harus disetel ke 14365123651274.
- x-amz-meta-tag Dapat berisi nilai apa pun.

Berikut ini adalah versi yang dikodekan dengan Base64 dari kebijakan ini.

```
eyAiZXhwaXJhdGlvbiI6ICIyMDA3LTEyLTAxVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXRpb25zIjogWwogICAgeyJidWN
```


Menggunakan kredensial Anda untuk membuat tanda tangan, misalnya `0RavWzkygo6QX9caELEqKi9kDbU=` adalah tanda tangan untuk dokumen kebijakan sebelumnya.

Formulir berikut ini mendukung permintaan POST ke bucket DOC-EXAMPLE-BUCKET yang menggunakan kebijakan ini.

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    ...
  </head>
  <body>
  ...
  <form action="https://DOC-EXAMPLE-BUCKET.s3.us-west-1.amazonaws.com/" method="post"
 enctype="multipart/form-data">
    Key to upload: <input type="input" name="key" value="user/eric/" /><br />
    <input type="hidden" name="acl" value="public-read" />
    <input type="hidden" name="success_action_redirect" value="https://
awsexamplebucket1.s3.us-west-1.amazonaws.com/successful_upload.html" />
    Content-Type: <input type="input" name="Content-Type" value="image/jpeg" /><br />
    <input type="hidden" name="x-amz-meta-uuid" value="14365123651274" />
    Tags for File: <input type="input" name="x-amz-meta-tag" value="" /><br />
    <input type="hidden" name="AWSAccessKeyId" value="AKIAIOSFODNN7EXAMPLE" />
    <input type="hidden" name="Policy" value="POLICY" />
    <input type="hidden" name="Signature" value="SIGNATURE" />
    File: <input type="file" name="file" /> <br />
    <!-- The elements after this will be ignored -->
    <input type="submit" name="submit" value="Upload to Amazon S3" />
  </form>
  ...
</html>
```

Permintaan sampel

Permintaan ini mengasumsikan bahwa gambar yang diunggah adalah 117,108 byte; data gambar tidak disertakan.

```
POST / HTTP/1.1
Host: awsexamplebucket1.s3.us-west-1.amazonaws.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.10) Gecko/20071115
 Firefox/2.0.0.10
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/
plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Content-Type: multipart/form-data; boundary=9431149156168
Content-Length: 118698

--9431149156168
Content-Disposition: form-data; name="key"

user/eric/MyPicture.jpg
--9431149156168
Content-Disposition: form-data; name="acl"

public-read
--9431149156168
Content-Disposition: form-data; name="success_action_redirect"

https://awsexamplebucket1.s3.us-west-1.amazonaws.com/successful_upload.html
--9431149156168
Content-Disposition: form-data; name="Content-Type"

image/jpeg
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-uuid"

14365123651274
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-tag"

Some,Tag,For,Picture
--9431149156168
Content-Disposition: form-data; name="AWSAccessKeyId"

AKIAIOSFODNN7EXAMPLE
--9431149156168
```

```
Content-Disposition: form-data; name="Policy"

eyAiZXhwaXJhdGlvbiI6ICIyMDA3LTEyLTAxVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXRpb25zIjogWwogICAgeyJidWN
--9431149156168
Content-Disposition: form-data; name="Signature"

0RavWzkygo6QX9caELEqKi9kDbU=
--9431149156168
Content-Disposition: form-data; name="file"; filename="MyFilename.jpg"
Content-Type: image/jpeg

...file content...
--9431149156168
Content-Disposition: form-data; name="submit"

Upload to Amazon S3
--9431149156168--
```

Contoh respons

```
HTTP/1.1 303 Redirect
x-amz-request-id: 1AEE782442F35865
x-amz-id-2: cxzFLJRatFHy+NGtaDFRR8YvI9BHmgLxjvJzNiGGICARZ/mVXHj7T+qQKhdpzHFh
Content-Type: application/xml
Date: Wed, 14 Nov 2007 21:21:33 GMT
Connection: close
Location: https://awsexamplebucket1.s3.us-west-1.amazonaws.com/
successful_upload.html?bucket=awsexamplebucket1&key=user/eric/
MyPicture.jpg&etag=&quot;39d459dfbc0faabbb5e179358dfb94c3&quot;
Server: AmazonS3
```

Unggahan bidang teks

Topik

- Pembuatan kebijakan dan formulir
- Permintaan sampel
- Contoh respons

Contoh berikut menunjukkan proses yang lengkap untuk membuat kebijakan dan formulir untuk mengunggah bidang teks. Mengunggah bidang teks berguna untuk mengirimkan konten yang dibuat pengguna, misalnya posting blog.

Pembuatan kebijakan dan formulir

Kebijakan berikut mendukung unggahan bidang teks ke Amazon S3 untuk bucket awsexamplebucket1.

```
{ "expiration": "2007-12-01T12:00:00.000Z",
  "conditions": [
    {"bucket": "awsexamplebucket1"},
    ["starts-with", "$key", "user/eric/"],
    {"acl": "public-read"},
    {"success_action_redirect": "https://awsexamplebucket1.s3.us-west-1.amazonaws.com/
new_post.html"},
    ["eq", "$Content-Type", "text/html"],
    {"x-amz-meta-uuid": "14365123651274"},
    ["starts-with", "$x-amz-meta-tag", ""]
  ]
}
```

Kebijakan ini mengharuskan:

- Unggahan harus dilakukan sebelum pukul 12:00 GMT pada tanggal 1 Desember 2007.
- Konten harus diunggah ke bucket awsexamplebucket1.
- Kunci harus dimulai dengan "user/eric/".
- ACL diatur menjadi baca publik.
- Success_action_redirect diatur ke https://awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html.
- Objek adalah teks HTML.
- x-amz-meta-uuid Tag harus disetel ke 14365123651274.
- x-amz-meta-tag Dapat berisi nilai apa pun.

Berikut ini adalah versi pengodean Base64 dari kebijakan ini.

```
eyAiZXhwaXJhdGlvbiI6ICIyMDA3LTEyLTAxVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXR
```

```
pb25zIjogWwogICAgeyJidWNrZXQiOiAiam9obnNtaXRoIn0sCiAgICBbInN0YXJ0cy13aXRoIiwgIiRrZXkiLCAidXNlci
LAogICAgeyJhY2wiOiAicHVibGljLXJlYWQifSwKICAgIHsic3VjY2Vzc19hY3Rpb25fcmVkaXJlY3QiOiAiaHR0cDovL2p
C5zMy5hbWF6b25hd3MuY29tL25ld19wb3N0LmH0bWwifSwKICAgIFsiZXEiLCAiJENvbnRlbnQtVHlwZSIsICJ0ZXh0L2h0
CAgIHsieC1hbXotbWV0YS11dWlkIjogIjE0MzY1MTIzNjUxMjc0In0sCiAgICBbInN0YXJ0cy13aXRoIiwgIiR4LWFtei1t
IsICIiXQogIF0KfQo=
```


Dengan menggunakan kredensial Anda, buat tanda tangan. Contohnya, `qA7FWXKq6VvU68lI9KdveT1cWgF=` adalah tanda tangan untuk dokumen kebijakan sebelumnya.

Formulir berikut ini mendukung permintaan POST ke bucket DOC-EXAMPLE-BUCKET yang menggunakan kebijakan ini.

```
<html>
  <head>
    ...
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    ...
  </head>
  <body>
  ...
  <form action="https://DOC-EXAMPLE-BUCKET.s3.us-west-1.amazonaws.com/" method="post"
 enctype="multipart/form-data">
    Key to upload: <input type="input" name="key" value="user/eric/" /><br />
    <input type="hidden" name="acl" value="public-read" />
    <input type="hidden" name="success_action_redirect" value="https://
awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html" />
    <input type="hidden" name="Content-Type" value="text/html" />
    <input type="hidden" name="x-amz-meta-uuid" value="14365123651274" />
    Tags for File: <input type="input" name="x-amz-meta-tag" value="" /><br />
    <input type="hidden" name="AWSAccessKeyId" value="AKIAIOSFODNN7EXAMPLE" />
    <input type="hidden" name="Policy" value="POLICY" />
    <input type="hidden" name="Signature" value="SIGNATURE" />
    Entry: <textarea name="file" cols="60" rows="10">

Your blog post goes here.

    </textarea><br />
    <!-- The elements after this will be ignored -->
    <input type="submit" name="submit" value="Upload to Amazon S3" />
  </form>
  ...
</html>
```

Permintaan sampel

Permintaan ini mengasumsikan bahwa gambar yang diunggah adalah 117,108 byte; data gambar tidak disertakan.

```
POST / HTTP/1.1
Host: awsexamplebucket1.s3.us-west-1.amazonaws.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.8.1.10) Gecko/20071115
 Firefox/2.0.0.10
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/
plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Content-Type: multipart/form-data; boundary=178521717625888
Content-Length: 118635

-178521717625888
Content-Disposition: form-data; name="key"

ser/eric/NewEntry.html
--178521717625888
Content-Disposition: form-data; name="acl"

public-read
--178521717625888
Content-Disposition: form-data; name="success_action_redirect"

https://awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html
--178521717625888
Content-Disposition: form-data; name="Content-Type"

text/html
--178521717625888
Content-Disposition: form-data; name="x-amz-meta-uuid"

14365123651274
--178521717625888
Content-Disposition: form-data; name="x-amz-meta-tag"

Interesting Post
--178521717625888
```

```
Content-Disposition: form-data; name="AWSAccessKeyId"

AKIAIOSFODNN7EXAMPLE
--178521717625888
Content-Disposition: form-data; name="Policy"
eyAiZXhwaXJhdGlvbiI6ICIyMDA3LTEyLTAxVDEyOjAwOjAwLjAwMFoiLAogICJjb25kaXRpb25zIjogWwogICAgeyJidWN

--178521717625888
Content-Disposition: form-data; name="Signature"

qA7FWXKq6VvU68lI9KdveT1cWgF=
--178521717625888
Content-Disposition: form-data; name="file"

...content goes here...
--178521717625888
Content-Disposition: form-data; name="submit"

Upload to Amazon S3
--178521717625888--
```

Contoh respons

```
HTTP/1.1 303 Redirect
x-amz-request-id: 1AEE782442F35865
x-amz-id-2: cxzFLJRatFHy+NGtaDFRR8YvI9BHmgLxjvJzNiGGICARZ/mVXHj7T+qQKhdpzHFh
Content-Type: application/xml
Date: Wed, 14 Nov 2007 21:21:33 GMT
Connection: close
Location: https://awsexamplebucket1.s3.us-west-1.amazonaws.com/new_post.html?
bucket=awsexamplebucket1&key=user/eric/
NewEntry.html&etag=40c3271af26b7f1672e41b8a274d28d4
Server: AmazonS3
```

POST dengan adobe flash

Bagian ini menjelaskan cara menggunakan `POST` dengan Adobe Flash.

Keamanan Adobe flash player

Secara default, model keamanan Adobe Flash Player akan melarang Adobe Flash Player membuat koneksi jaringan ke server di luar domain yang melayani file SWF.

Untuk menimpa pengaturan default, Anda harus mengunggah file crossdomain.xml yang dapat dibaca publik ke bucket yang akan menerima unggahan POST. Berikut ini adalah sampel dari file crossdomain.xml.

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM
"http://www.macromedia.com/xml/dtds/cross-domain-policy.dtd">
<cross-domain-policy>
<allow-access-from domain="*" secure="false" />
</cross-domain-policy>
```

> **Note**
>
> Untuk informasi selengkapnya tentang model keamanan Adobe Flash, kunjungi situs web Adobe.
>
> Menambahkan file crossdomain.xml ke bucket akan membuat Adobe Flash Player terhubung ke file crossdomain.xml dalam bucket Anda; namun, file ini tidak akan memberikan akses ke bucket Amazon S3 sebenarnya.

Pertimbangan Adobe flash

FileReference API di Adobe Flash menambahkan bidang `Filename` formulir ke permintaan POST. Saat Anda membuat aplikasi Adobe Flash yang mengunggah ke Amazon S3 menggunakan tindakan FileReference API, sertakan kondisi berikut dalam kebijakan Anda:

```
['starts-with', '$Filename', '']
```

Beberapa versi Adobe Flash Player tidak menangani dengan benar respons HTTP yang memiliki bodi kosong. Untuk mengonfigurasi POST guna mengembalikan respons yang tidak memiliki bodi kosong, atur `success_action_status` menjadi 201. Amazon S3 kemudian akan mengembalikan dokumen XML dengan kode status 201. Untuk informasi tentang konten dokumen XML, lihat Objek POST. Untuk informasi tentang kolom formulir, lihat Bidang formulir HTML.

Pola desain praktik terbaik: mengoptimalkan performa Amazon S3

Aplikasi Anda dapat dengan mudah mencapai ribuan transaksi per detik dalam performa permintaan saat mengunggah dan mengambil penyimpanan dari Amazon S3. Amazon S3 secara otomatis menyesuaikan ke tingkat permintaan yang tinggi. Misalnya, aplikasi Anda dapat mencapai setidaknya 3.500 permintaan PUT/COPY/POST/DELETE, atau 5.500 permintaan GET/HEAD per detik per prefiks Amazon S3 yang dipartisi. Tidak ada batas jumlah prefiks dalam bucket. Anda dapat meningkatkan performa membaca atau menulis dengan menggunakan paralelisasi. Misalnya, jika Anda membuat 10 prefiks dalam bucket Amazon S3 untuk paralelisasi bacaan, Anda dapat menskalakan performa baca Anda menjadi 55.000 permintaan baca per detik. Demikian pula, Anda dapat menskalakan operasi tulis dengan menulis ke beberapa prefiks. Penskalaan, dalam kasus operasi baca dan tulis, terjadi secara bertahap dan tidak instan. Sementara Amazon S3 menskalakan ke tingkat permintaan baru Anda yang lebih tinggi, Anda mungkin melihat beberapa kesalahan 503 (Perlambat). Kesalahan ini akan hilang saat penskalaan selesai. Untuk informasi selengkapnya tentang cara membuat dan menggunakan prefiks, lihat Organisasi objek menggunakan prefiks.

Beberapa aplikasi danau data di Amazon S3 memindai jutaan atau miliar objek untuk kueri yang berjalan di atas petabyte data. Aplikasi danau data ini mencapai laju transfer jarak tunggal yang memaksimalkan penggunaan antarmuka jaringan untuk instans Amazon EC2, yang dapat mencapai 100 Gb/s dalam satu instans. Aplikasi ini kemudian menggabungkan throughput di beberapa instans untuk mendapatkan beberapa terabit per detik.

Aplikasi lain sensitif terhadap latensi, seperti aplikasi perpesanan media sosial. Aplikasi ini dapat mencapai latensi objek kecil yang konsisten (dan first-byte-out latensi untuk objek yang lebih besar) sekitar 100-200 milidetik.

AWS Layanan lain juga dapat membantu mempercepat kinerja untuk arsitektur aplikasi yang berbeda. Misalnya, jika Anda menginginkan kecepatan transfer yang lebih tinggi melalui koneksi HTTP tunggal atau latensi milidetik satu digit, gunakan Amazon atau Amazon ElastiCache untuk caching dengan CloudFront Amazon S3.

Selain itu, jika Anda ingin transportasi data cepat melalui jarak jauh antara klien dan bucket S3, gunakan Mengonfigurasi transfer file yang cepat dan aman menggunakan Amazon S3 Transfer Acceleration. Transfer Acceleration menggunakan lokasi edge yang didistribusikan secara global CloudFront untuk mempercepat transportasi data melalui jarak geografis. Jika beban kerja Amazon S3 Anda menggunakan enkripsi sisi server AWS KMS, lihat AWS KMS Batas di Panduan AWS Key

Management Service Pengembang untuk informasi tentang tarif permintaan yang didukung untuk kasus penggunaan Anda.

Topik berikut menjelaskan panduan praktik terbaik dan pola desain untuk mengoptimalkan performa aplikasi yang menggunakan Amazon S3. Lihat Pedoman Kinerja untuk Amazon S3 dan Pola Desain Performa untuk Amazon S3 untuk informasi terkini tentang optimalisasi performa untuk Amazon S3.

> **Note**
>
> Untuk informasi selengkapnya tentang menggunakan kelas penyimpanan Amazon S3 Express One Zone dengan bucket direktori, lihat Apa itu S3 Express One Zone? dan Ember direktori.

Topik

- Pedoman Kinerja untuk Amazon S3
- Pola Desain Performa untuk Amazon S3

Pedoman Kinerja untuk Amazon S3

Ketika membangun aplikasi yang mengunggah dan mengambil objek dari Amazon S3, ikuti panduan praktik terbaik kami untuk mengoptimalkan performa. Kami juga menawarkan Pola Desain Performa yang lebih terperinci.

Untuk mendapatkan performa terbaik untuk aplikasi Anda di Amazon S3, kami merekomendasikan panduan berikut ini.

Topik

- Mengukur Performa
- Skala Koneksi Penyimpanan Secara Horizontal
- Menggunakan Byte-Range Fetches
- Permintaan Coba Lagi untuk Aplikasi yang Sensitif Latensi
- Gabungkan Amazon S3 (Penyimpanan) dan Amazon EC2 (Komputasi) yang Sama Wilayah AWS
- Gunakan Amazon S3 Transfer Acceleration untuk Meminimalkan Latensi yang Disebabkan oleh Jarak

- Gunakan versi terbaru dari SDK AWS

Mengukur Performa

Ketika mengoptimalkan performa, lihat kebutuhan throughput jaringan, CPU, dan DRAM. Bergantung pada perpaduan permintaan untuk sumber daya yang berbeda ini, mungkin akan untuk mengevaluasi berbagai jenis instans Amazon EC2. Untuk informasi selengkapnya tentang jenis instans, lihat Jenis Instans di Panduan Pengguna Amazon EC2.

Ini juga membantu untuk melihat waktu pencarian DNS, latensi, dan kecepatan transfer data menggunakan alat analisis HTTP saat mengukur performa.

Untuk memahami persyaratan performa dan mengoptimalkan performa aplikasi Anda, Anda juga dapat memantau respons kesalahan 503 yang Anda terima. Memantau metrik performa tertentu dapat menimbulkan biaya tambahan. Untuk informasi selengkapnya, lihat Harga Amazon S3.

Pantau jumlah respons kesalahan status 503 (Perlambat)

Untuk memantau jumlah respons kesalahan status 503 yang Anda dapatkan, Anda dapat menggunakan salah satu opsi berikut:

- Gunakan metrik CloudWatch permintaan Amazon untuk Amazon S3. Metrik CloudWatch permintaan menyertakan metrik untuk respons status 5xx. Untuk informasi selengkapnya tentang metrik CloudWatch permintaan, lihatMemantau metrik dengan Amazon CloudWatch.
- Gunakan jumlah kesalahan 503 (Layanan Tidak Tersedia) yang tersedia di bagian metrik lanjutan Lensa Penyimpanan Amazon S3. Untuk informasi selengkapnya, lihat Menggunakan metrik S3 Storage Lens untuk meningkatkan kinerja.
- Gunakan pencatatan log akses server Amazon S3. Dengan pencatatan akses server, Anda dapat memfilter dan meninjau semua permintaan yang menerima respons 503 (Kesalahan Internal). Anda juga dapat menggunakan Amazon Athena untuk mengurai log. Untuk informasi selengkapnya tentang pencatatan log akses server, lihat Pencatatan permintaan dengan pencatatan akses server.

Dengan memantau jumlah kode kesalahan status HTTP 503, Anda sering kali dapat memperoleh wawasan berharga tentang prefiks, kunci, atau bucket mana yang mendapatkan permintaan throttling paling banyak.

Skala Koneksi Penyimpanan Secara Horizontal

Menyebarkan permintaan di berbagai koneksi adalah pola desain umum untuk menskalakan performa secara horizontal. Ketika Anda membangun aplikasi berperforma tinggi, bayangkan Amazon S3 sebagai sistem terdistribusi yang sangat besar, bukan sebagai titik akhir jaringan tunggal seperti server penyimpanan tradisional. Anda dapat mencapai performa terbaik dengan mengeluarkan beberapa permintaan sekaligus ke Amazon S3. Sebarkan permintaan ini melalui koneksi terpisah untuk memaksimalkan bandwidth yang dapat diakses dari Amazon S3. Amazon S3 tidak memiliki batas untuk jumlah koneksi yang dilakukan ke bucket Anda.

Menggunakan Byte-Range Fetches

Menggunakan Header HTTP `Range` di permintaan GET Object, Anda dapat mengambil rentang byte dari objek, dengan hanya mentransfer bagian tertentu. Anda dapat menggunakan koneksi bersamaan ke Amazon S3 untuk mengambil byte yang berbeda dari dalam objek yang sama. Ini membantu Anda mencapai throughput agregat yang lebih tinggi dibandingkan permintaan objek utuh tunggal. Pengambilan rentang objek besar yang lebih kecil juga memungkinkan aplikasi Anda untuk meningkatkan waktu coba kembali ketika permintaan terganggu. Untuk informasi selengkapnya, lihat Mengunggah objek.

Ukuran umum untuk permintaan rentang byte adalah 8 MB atau 16 MB. Jika objek adalah PUT yang menggunakan pengunggahan multibagian, praktik yang baik adalah GET dalam ukuran bagian yang sama (atau setidaknya sejajar dengan batas bagian) untuk performa terbaik. Permintaan GET dapat secara langsung menangani bagian individual; misalnya, `GET ?partNumber=N`.

Permintaan Coba Lagi untuk Aplikasi yang Sensitif Latensi

Batas waktu dan percobaan ulang yang agresif membantu mendorong latensi yang konsisten. Mengingat skala besar Amazon S3, jika permintaan pertama berjalan lambat, permintaan yang dicoba kembali kemungkinan besar mengambil jejak yang berbeda dan dengan cepat berhasil. AWS SDK memiliki batas waktu yang dapat dikonfigurasi dan nilai coba ulang yang dapat Anda sesuaikan dengan toleransi aplikasi spesifik Anda.

Gabungkan Amazon S3 (Penyimpanan) dan Amazon EC2 (Komputasi) yang Sama Wilayah AWS

Meskipun nama bucket S3 bersifat unik secara global, setiap bucket disimpan di Wilayah yang Anda pilih saat membuat bucket. Untuk mengoptimalkan kinerja, sebaiknya Anda mengakses bucket

dari instans Amazon EC2 secara bersamaan Wilayah AWS jika memungkinkan. Ini membantu mengurangi latensi jaringan dan biaya transfer data.

Untuk informasi lebih lanjut tentang biaya transfer data, lihat Harga Amazon S3.

Gunakan Amazon S3 Transfer Acceleration untuk Meminimalkan Latensi yang Disebabkan oleh Jarak

Mengonfigurasi transfer file yang cepat dan aman menggunakan Amazon S3 Transfer Acceleration mengelola transfer file yang cepat, mudah, dan aman dalam jarak geografis yang panjang antara klien dan bucket S3. Transfer Acceleration memanfaatkan lokasi edge yang didistribusikan secara global di Amazon CloudFront. Saat data tiba di lokasi edge, data diarahkan ke Amazon S3 melalui jalur jaringan yang dioptimalkan. Transfer Acceleration ideal untuk mentransfer gigabyte ke terabyte data secara teratur melintasi benua. Ini juga berguna bagi klien yang mengunggah ke bucket terpusat dari seluruh dunia.

Anda dapat menggunakan Amazon S3 Transfer Acceleration Speed Comparison tool untuk membandingkan kecepatan unggahan yang dipercepat dan tidak dipercepat di seluruh Wilayah Amazon S3. Alat Speed Comparison menggunakan unggahan multibagian untuk mentransfer file dari browser Anda ke berbagai Wilayah Amazon S3 dengan dan tanpa menggunakan Amazon S3 Transfer Acceleration.

Gunakan versi terbaru dari SDK AWS

AWS SDK menyediakan dukungan bawaan untuk banyak pedoman yang direkomendasikan untuk mengoptimalkan kinerja Amazon S3. SDK menyediakan API yang lebih sederhana untuk memanfaatkan Amazon S3 dari dalam aplikasi dan diperbarui secara berkala untuk mengikuti praktik terbaik terbaru. Misalnya, SDK memasukkan logika untuk secara otomatis mencoba kembali permintaan pada kesalahan HTTP 503 dan berinvestasi dalam kode untuk merespons dan beradaptasi dengan koneksi yang lambat.

SDK juga menyediakan Transfer Manager, yang mengotomatiskan koneksi penskalaan horizontal untuk mencapai ribuan permintaan per detik, menggunakan permintaan rentang byte di mana itu sesuai. Penting untuk menggunakan AWS SDK versi terbaru untuk mendapatkan fitur pengoptimalan kinerja terbaru.

Anda juga dapat mengoptimalkan performa saat Anda menggunakan permintaan API REST HTTP. Saat menggunakan API REST, Anda harus mengikuti praktik terbaik yang sama yang merupakan

bagian dari SDK. Izinkan batas waktu dan percobaan ulang pada permintaan yang lambat, dan beberapa koneksi untuk memungkinkan pengambilan data objek secara paralel. Untuk informasi selengkapnya tentang API REST, lihat Referensi API Amazon Simple Storage Service.

Pola Desain Performa untuk Amazon S3

Saat merancang aplikasi untuk mengunggah dan mengambil objek dari Amazon S3, gunakan pola desain praktik terbaik kami untuk mencapai performa terbaik untuk aplikasi Anda. Kami juga menawarkan Pedoman Kinerja untuk Anda pertimbangkan saat sedang merencanakan arsitektur aplikasi Anda.

Untuk mengoptimalkan performa, Anda dapat menggunakan pola desain berikut.

Topik

- Menggunakan Caching untuk Konten yang Sering Diakses
- Batas Waktu dan Pengulangan untuk Aplikasi yang Sensitif-Latensi
- Penskalaan Horizontal dan Paralelisasi Permintaan untuk Throughput Tinggi
- Menggunakan Amazon S3 Transfer Acceleration untuk Mempercepat Transfer Data Terpisah Secara Geografis

Menggunakan Caching untuk Konten yang Sering Diakses

Banyak aplikasi yang menyimpan data di Amazon S3 melayani "perangkat kerja” data yang berulang kali diminta oleh pengguna. Jika beban kerja mengirimkan permintaan GET berulang untuk sekumpulan objek umum, Anda dapat menggunakan cache seperti Amazon CloudFront, Amazon ElastiCache, atau AWS Elemental MediaStoreuntuk mengoptimalkan kinerja. Penerapan cache yang berhasil dapat menghasilkan latensi rendah dan laju transfer data tinggi. Aplikasi yang menggunakan caching juga mengirimkan lebih sedikit permintaan langsung ke Amazon S3, yang dapat membantu mengurangi biaya permintaan.

Amazon CloudFront adalah jaringan pengiriman konten cepat (CDN) yang secara transparan menyimpan data dari Amazon S3 dalam satu set besar titik kehadiran yang didistribusikan secara geografis (). PoPs Ketika objek dapat diakses dari beberapa Wilayah, atau melalui internet, CloudFront memungkinkan data untuk di-cache dekat dengan pengguna yang mengakses objek. Ini dapat menghasilkan pengiriman konten Amazon S3 populer dengan performa tinggi. Untuk selengkapnya CloudFront, lihat Panduan CloudFront Pengembang Amazon.

Amazon ElastiCache adalah cache dalam memori yang dikelola. Dengan ElastiCache, Anda dapat menyediakan instans Amazon EC2 yang menyimpan objek cache dalam memori. Hasil caching ini menghasilkan urutan pengurangan besar-besaran dalam latensi GET dan peningkatan substansial dalam throughput unduhan. Untuk menggunakannya ElastiCache, Anda memodifikasi logika aplikasi untuk mengisi cache dengan objek panas dan memeriksa cache untuk objek panas sebelum memintanya dari Amazon S3. Untuk contoh penggunaan ElastiCache untuk meningkatkan kinerja Amazon S3 GET, lihat posting blog Turbocharge Amazon S3 dengan Amazon untuk Redis. ElastiCache

AWS Elemental MediaStore adalah sistem caching dan distribusi konten yang khusus dibuat untuk alur kerja video dan pengiriman media dari Amazon S3. MediaStore menyediakan API end-to-end penyimpanan khusus untuk video, dan direkomendasikan untuk beban kerja video yang sensitif terhadap kinerja. Untuk selengkapnya MediaStore, lihat Panduan AWS Elemental MediaStore Pengguna.

Batas Waktu dan Pengulangan untuk Aplikasi yang Sensitif-Latensi

Ada situasi tertentu ketika aplikasi menerima respons dari Amazon S3 yang menunjukkan bahwa percobaan ulang diperlukan. Bucket peta Amazon S3 dan nama objek ke data objek yang terkait dengannya. Jika aplikasi menghasilkan tingkat permintaan yang tinggi (biasanya tingkat permintaan lebih dari 5.000 permintaan per detik ke sejumlah kecil objek), aplikasi mungkin akan menerima respons perlambatan HTTP 503. Jika kesalahan ini terjadi, setiap SDK AWS menerapkan logika percobaan ulang otomatis menggunakan mundur eksponensial. Jika Anda tidak menggunakan SDK AWS , Anda harus menerapkan logika percobaan ulang saat menerima kesalahan HTTP 503. Untuk informasi tentang teknik back-off, lihat Error Retries and Exponential Backoff di. AWSReferensi Umum Amazon Web Services

Amazon S3 secara otomatis menskalakan dalam menanggapi tingkat permintaan baru yang berkelanjutan, secara dinamis mengoptimalkan performa. Meskipun Amazon S3 secara internal mengoptimalkan untuk tingkat permintaan baru, Anda akan menerima respons permintaan HTTP 503 secara sementara hingga optimalisasi selesai. Setelah Amazon S3 secara internal mengoptimalkan performa untuk tingkat permintaan baru, semua permintaan umumnya dilayani tanpa percobaan ulang.

Untuk aplikasi sensitif latensi, Amazon S3 menyarankan pelacakan dan mencoba kembali operasi yang lebih lambat secara agresif. Saat Anda mencoba lagi permintaan, kami menyarankan menggunakan koneksi baru ke Amazon S3 dan melakukan pencarian DNS baru.

Saat Anda membuat permintaan berukuran besar (misalnya, lebih dari 128 MB), kami menyarankan untuk melacak throughput yang dicapai, dan mencoba kembali 5 persen paling lambat dari permintaan tersebut. Saat Anda membuat permintaan yang lebih kecil (misalnya, kurang dari 512 KB), yaitu latensi median sering kali berada dalam rentang milidetik, panduan yang baik adalah mencoba kembali operasi GET atau PUT setelah 2 detik. Jika percobaan ulang tambahan diperlukan, praktik terbaik adalah back off. Misalnya, kami sarankan untuk mengeluarkan satu percobaan ulang setelah 2 detik dan percobaan ulang kedua setelah 4 detik tambahan.

Jika aplikasi Anda membuat permintaan ukuran tetap ke Amazon S3, Anda akan mengharapkan waktu respons yang lebih konsisten untuk setiap permintaan ini. Dalam kasus ini, strategi sederhana adalah mengidentifikasi 1 persen permintaan paling lambat dan mencobanya lagi. Bahkan, satu kali coba ulang sering kali efektif dalam mengurangi latensi.

Jika Anda menggunakan AWS Key Management Service (AWS KMS) untuk enkripsi sisi server, lihat Batas dalam Panduan AWS Key Management Service Pengembang untuk informasi tentang tarif permintaan yang didukung untuk kasus penggunaan Anda.

Penskalaan Horizontal dan Paralelisasi Permintaan untuk Throughput Tinggi

Amazon S3 adalah sistem terdistribusi yang sangat besar. Untuk membantu Anda memanfaatkan skalanya, kami mendorong Anda untuk menskalakan permintaan paralel secara horizontal ke titik akhir layanan Amazon S3. Selain mendistribusikan permintaan di dalam Amazon S3, jenis pendekatan penskalaan ini membantu mendistribusikan muatan melalui beberapa jejak melalui jaringan.

Untuk transfer dengan throughput tinggi, Amazon S3 menyarankan penggunaan aplikasi yang menggunakan beberapa koneksi untuk GET atau PUT data secara paralel. Misalnya, ini didukung oleh Amazon S3 Transfer Manager di AWS Java SDK, dan sebagian besar AWS SDK lainnya menyediakan konstruksi serupa. Untuk beberapa aplikasi, Anda dapat mencapai koneksi paralel dengan meluncurkan beberapa permintaan secara bersamaan dalam utas aplikasi yang berbeda, atau dalam instans aplikasi yang berbeda. Pendekatan terbaik untuk diambil tergantung pada aplikasi Anda dan struktur objek yang Anda akses.

Anda dapat menggunakan AWS SDK untuk mengeluarkan permintaan GET dan PUT secara langsung daripada menggunakan manajemen transfer di SDK AWS . Dengan pendekatan ini, Anda dapat mengatur beban kerja secara lebih langsung, sembari tetap memanfaatkan dukungan SDK untuk percobaan ulang dan penanganannya terhadap setiap respons HTTP 503 yang mungkin

terjadi. Sebagai aturan umum, ketika Anda mengunduh objek besar di dalam Wilayah dari Amazon S3 ke Amazon EC2, kami menyarankan untuk membuat permintaan bersamaan untuk kisaran byte objek pada granularitas sebesar 8–16 MB. Ajukan satu permintaan bersamaan untuk setiap 85–90 MB/dtk dari throughput jaringan yang diinginkan. Untuk memenuhi kartu antarmuka jaringan (NIC) 10 Gb/dtk, Anda mungkin menggunakan sekitar 15 permintaan secara bersamaan melalui koneksi terpisah. Anda dapat menaikkan skala permintaan serentak melalui lebih banyak koneksi untuk mensaturasikan NIC yang lebih cepat, seperti NIC 25 Gb/dtk atau 100 Gb/dtk.

Mengukur performa sangat penting saat Anda menyesuaikan jumlah permintaan untuk masalah secara bersamaan. Kami menyarankan untuk memulai dengan permintaan tunggal pada satu waktu. Mengukur bandwith jaringan yang dicapai dan penggunaan sumber daya lain yang digunakan aplikasi Anda dalam memproses data. Anda kemudian dapat mengidentifikasi sumber daya bottleneck (yaitu, sumber daya dengan penggunaan tertinggi), dan karenanya jumlah permintaan yang mungkin berguna. Misalnya, jika memproses satu permintaan pada satu waktu menghasilkan penggunaan CPU sebesar 25 persen, hal ini menunjukkan bahwa hingga empat permintaan bersamaan dapat diakomodasi. Pengukuran sangat penting, dan perlu mengonfirmasi penggunaan sumber daya karena tingkat permintaan meningkat.

Jika aplikasi Anda menerbitkan permintaan secara langsung ke Amazon S3 menggunakan API REST, kami menyarankan menggunakan kumpulan koneksi HTTP dan menggunakan kembali setiap koneksi untuk serangkaian permintaan. Menghindari penyiapan koneksi per permintaan menghilangkan kebutuhan untuk menjalankan TCP slow-start dan handshake Secure Sockets Layer (SSL) pada setiap permintaan. Untuk informasi selengkapnya tentang API REST, lihat Referensi API Amazon Simple Storage Service.

Terakhir, penting untuk memperhatikan DNS dan pemeriksaan ulang bahwa permintaan tersebar di sejumlah besar alamat IP Amazon S3. Kueri DNS untuk siklus Amazon S3 melalui daftar besar titik akhir IP. Namun pemecah cache atau kode aplikasi yang menggunakan kembali satu alamat IP tidak mendapatkan keuntungan dari keragaman alamat dan load balancing yang mengikutinya. Alat utilitas jaringan seperti alat baris perintah `netstat` dapat menunjukkan alamat IP yang digunakan untuk komunikasi dengan Amazon S3, dan kami memberikan panduan untuk konfigurasi DNS untuk digunakan. Untuk informasi selengkapnya tentang pedoman ini, lihat Membuat permintaan.

Menggunakan Amazon S3 Transfer Acceleration untuk Mempercepat Transfer Data Terpisah Secara Geografis

Mengonfigurasi transfer file yang cepat dan aman menggunakan Amazon S3 Transfer Acceleration terbukti efektif dalam meminimalkan atau menghilangkan latensi yang disebabkan oleh jarak

geografis antara klien yang tersebar secara global dan aplikasi regional menggunakan Amazon S3. Transfer Acceleration menggunakan lokasi edge yang didistribusikan secara global CloudFront untuk transportasi data. Jaringan AWS edge memiliki titik kehadiran di lebih dari 50 lokasi. Saat ini, digunakan untuk mendistribusikan konten melalui CloudFront dan untuk memberikan respons cepat terhadap kueri DNS yang dibuat ke Amazon Route 53.

Jaringan edge juga membantu mempercepat transfer data masuk dan keluar dari Amazon S3. Sangat ideal untuk aplikasi yang mentransfer data antar benua, memiliki koneksi internet yang cepat, menggunakan objek besar, atau memiliki banyak konten untuk diunggah. Saat datanya tiba di lokasi edge, data diarahkan ke Amazon S3 melalui jalur jaringan yang dioptimalkan. Secara umum, semakin jauh Anda berada di Wilayah Amazon S3, semakin tinggi peningkatan kecepatan yang dapat Anda harapkan dari penggunaan Transfer Acceleration.

Anda dapat mengatur Transfer Acceleration pada bucket baru atau yang sudah ada. Anda dapat menggunakan endpoint Amazon S3 Transfer Acceleration terpisah untuk menggunakan lokasi AWS tepi. Cara terbaik untuk menguji apakah Transfer Acceleration membantu klien meminta performa adalah menggunakan Alat Amazon S3 Transfer Acceleration Speed Comparison. Konfigurasi dan kondisi jaringan bervariasi dari waktu ke waktu dan dari lokasi ke lokasi. Jadi, Anda hanya dikenakan biaya untuk transfer ketika Amazon S3 Transfer Acceleration dapat meningkatkan performa pengunggahan Anda. Untuk informasi tentang penggunaan Akselerasi Transfer dengan AWS SDK yang berbeda, lihatMengaktifkan dan menggunakan S3 Transfer Acceleration.

Apa itu Amazon S3 di Outposts?

AWS Outposts adalah layanan yang dikelola sepenuhnya yang menawarkan AWS infrastruktur, AWS layanan, API, dan alat yang sama ke hampir semua pusat data, ruang co-lokasi, atau fasilitas lokal untuk pengalaman hybrid yang benar-benar konsisten. AWS Outposts sangat ideal untuk beban kerja yang memerlukan akses latensi rendah ke sistem lokal, pemrosesan data lokal, residensi data, dan migrasi aplikasi dengan saling ketergantungan sistem lokal. Untuk informasi selengkapnya, lihat Apa itu AWS Outposts? dalam AWS Outposts Panduan Pengguna.

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di Outposts Anda serta dengan mudah menyimpan dan mengambil objek on-premise. S3 di Outposts menyediakan kelas penyimpanan baru, `OUTPOSTS`, yang menggunakan API Amazon S3, serta dirancang untuk menyimpan data secara tahan lama dan berlebihan di beberapa perangkat dan server di Outposts Anda. Anda berkomunikasi dengan bucket Outposts menggunakan titik akses dan koneksi titik akhir melalui cloud privat virtual (VPC).

Anda dapat menggunakan API dan fitur yang sama di bucket Outposts seperti yang Anda lakukan di Amazon S3, termasuk kebijakan akses, enkripsi, dan pemberian tag. Anda dapat menggunakan S3 di Outposts melalui AWS Management Console AWS Command Line Interface ,AWS CLI(), SDK AWS , atau REST API.

- Cara kerja S3 di Outposts
- Fitur S3 di Outposts
- Layanan terkait
- Mengakses S3 di Outposts
- Membayar untuk S3 di Outposts
- Langkah selanjutnya

Cara kerja S3 di Outposts

S3 di Outposts adalah layanan penyimpanan objek yang menyimpan data sebagai objek dalam bucket di Outpost Anda. Objek adalah file data dan metadata opsional apa pun yang menjelaskan file tersebut. Bucket adalah kontainer untuk objek.

Untuk menyimpan data Anda di S3 di Outposts, Anda terlebih dahulu membuat bucket. Saat membuat bucket, Anda menentukan nama bucket dan Outpost yang akan menampung bucket. Untuk

mengakses bucket S3 di Outposts dan mengoperasikan objek, Anda selanjutnya membuat dan mengonfigurasikan titik akses. Anda juga harus membuat titik akhir untuk merutekan permintaan ke titik akses Anda.

Titik akses menyederhanakan akses data untuk aplikasi apa pun Layanan AWS atau pelanggan yang menyimpan data di S3. Titik akses diberi nama titik akhir jaringan yang melekat ke bucket serta dapat digunakan untuk melakukan operasi objek, seperti `GetObject` dan `PutObject`. Setiap titik akses memiliki izin dan kendali jaringan yang berbeda.

Anda dapat membuat dan mengelola S3 di bucket Outposts, titik akses, dan titik akhir dengan menggunakan AWS Management Console,, SDK AWS CLI AWS , atau REST API. Untuk mengunggah dan mengelola objek di bucket S3 di Outposts, Anda dapat menggunakan AWS CLI, SDK AWS , atau REST API.

Wilayah

Selama AWS Outposts penyediaan, Anda atau AWS membuat koneksi tautan layanan yang menghubungkan Pos Luar Anda kembali ke Wilayah Outposts pilihan Anda atau Wilayah AWS Outposts untuk operasi bucket dan telemetri. Outposts mengandalkan konektivitas ke Wilayah AWS induk. Rak Outposts tidak dirancang untuk operasi atau lingkungan yang tidak terhubung dengan konektivitas terbatas atau tanpa konektivitas. Untuk informasi selengkapnya, lihat Konektivitas Outposts ke Wilayah AWS di AWS Outposts Panduan Pengguna.

Bucket

Bucket adalah kontainer untuk objek yang disimpan dalam S3 di Outposts. Anda dapat menyimpan berapa pun jumlah objek dalam bucket dan dapat memiliki hingga 100 bucket per akun per Outpost.

Saat Anda membuat bucket, Anda memasukkan nama bucket dan memilih Outpost tempat bucket akan berada. Setelah Anda membuat bucket, Anda tidak dapat mengubah nama bucket atau memindahkan bucket ke Outpost lain. Nama bucket harus mengikuti aturan penamaan bucket Amazon S3. Di S3 on Outposts, nama bucket unik untuk Outpost dan. Akun AWS Bucket S3 di Outposts memerlukan `outpost-id`, `account-id`, dan nama bucket untuk mengidentifikasinya.

Contoh berikut menunjukkan format Amazon Resource Name (ARN) untuk bucket S3 di Outposts. ARN terdiri atas Wilayah asal Outposts Anda, akun Outpost Anda, ID Outposts, dan nama bucket.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/bucket/bucket-name
```

Setiap objek dimuat dalam bucket. Anda harus menggunakan titik akses untuk mengakses objek apa pun dalam bucket Outposts. Saat menentukan bucket untuk operasi objek, Anda menggunakan ARN titik akses atau alias titik akses. Untuk informasi selengkapnya tentang alias titik akses, lihat Menggunakan alias untuk titik akses bucket S3 di Outposts Anda di bucket S3 di Outposts.

Contoh berikut menunjukkan format ARN titik akses untuk S3 di Outposts, yang mencakup `outpost-id`, `account-id`, dan nama titik akses:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Untuk informasi selengkapnya tentang bucket, lihat Bekerja dengan S3 di bucket Outposts.

Objek

Objek adalah entitas dasar yang disimpan di S3 di Outposts. Objek terdiri atas data objek dan metadata. Metadata adalah serangkaian pasangan nilai-nama yang menjelaskan objek. Pasangan ini mencakup beberapa metadata default, seperti tanggal terakhir diubah, dan metadata HTTP standar, seperti `Content-Type`. Anda juga dapat menentukan metadata kustom pada saat objek disimpan. Objek diidentifikasi secara unik dalam bucket dengan kunci (atau nama).

Dengan Amazon S3 di Outposts, data objek selalu disimpan di Outpost. Saat AWS memasang rak Outpost, data Anda tetap lokal di Outpost Anda untuk memenuhi persyaratan residensi data. Objek Anda tidak akan meninggalkan Outpost dan tidak berada di Wilayah AWS. Karena di-host In-region, Anda tidak dapat menggunakan konsol untuk mengunggah atau mengelola objek di Outpost Anda. AWS Management Console Namun, Anda dapat menggunakan REST API, AWS Command Line Interface (AWS CLI), dan AWS SDK untuk mengunggah dan mengelola objek Anda melalui titik akses Anda.

Kunci

Kunci objek (atau nama kunci) adalah pengidentifikasi unik untuk objek dalam bucket. Setiap objek dalam bucket memiliki persis satu kunci. Kombinasi bucket dan kunci objek secara unik mengidentifikasi setiap objek.

Contoh berikut menunjukkan format ARN untuk S3 pada objek Outposts, yang mencakup Wilayah AWS kode untuk Wilayah tempat Pos Luar ditempatkan, Akun AWS ID, ID Outpost, nama bucket, dan kunci objek:

```
arn:aws:s3-outposts:us-west-2:123456789012:outpost/ op-01ac5d28a6a232904/
bucket/example-s3-bucket1/object/myobject
```

Untuk informasi selengkapnya tentang kunci objek, lihat Bekerja dengan S3 di objek Outposts.

Penentuan Versi S3

Anda dapat menggunakan Penentuan Versi S3 di bucket Outposts untuk menyimpan beberapa varian objek dalam bucket yang sama. Dengan Penentuan Versi S3, Anda dapat menyimpan, mengambil, dan memulihkan setiap versi dari setiap objek yang disimpan dalam bucket Anda. Penentuan Versi S3 membantu Anda memulihkan dari tindakan pengguna yang tidak diinginkan dan kegagalan aplikasi.

Untuk informasi selengkapnya, lihat Mengelola versioning.

ID Versi

Saat Anda mengaktifkan Penentuan Versi S3 dalam bucket, S3 di Outposts menghasilkan ID versi unik untuk setiap objek yang ditambahkan ke bucket. Objek yang sudah ada di bucket pada saat Anda mengaktifkan Penentuan Versi memiliki ID versi `null`. Jika Anda memodifikasi objek ini (atau lainnya) dengan operasi lain PutObject, seperti, objek baru mendapatkan ID versi unik.

Untuk informasi selengkapnya, lihat Mengelola versioning.

Kelas penyimpanan dan enkripsi

S3 di Outposts menyediakan kelas penyimpanan baru, S3 Outposts (`OUTPOSTS`). Kelas penyimpanan S3 Outposts hanya tersedia untuk objek yang disimpan dalam bucket di AWS Outposts. Jika Anda mencoba menggunakan kelas penyimpanan S3 lainnya dengan S3 pada Outposts, S3 di Outposts mengembalikan kesalahan `InvalidStorageClass` tersebut.

Secara default, objek yang disimpan dalam kelas penyimpanan S3 Outposts (`OUTPOSTS`) disimpan menggunakan enkripsi di sisi server dengan kunci enkripsi yang dikelola Amazon S3 (SSE-S3). Untuk informasi selengkapnya, lihat Enkripsi data di S3 di Outposts.

Kebijakan bucket

Kebijakan bucket adalah kebijakan berbasis sumber daya AWS Identity and Access Management (IAM) yang dapat Anda gunakan untuk memberikan izin akses ke bucket dan objek di dalamnya.

Hanya pemilik bucket yang dapat mengaitkan kebijakan dengan bucket. Izin yang dipasang pada bucket berlaku untuk semua objek di bucket yang dimiliki oleh pemilik bucket. Kebijakan bucket dibatasi hingga ukuran 20 KB.

Kebijakan bucket menggunakan bahasa kebijakan IAM berbasis JSON yang standar di seluruh AWS. Anda dapat menggunakan kebijakan bucket untuk menambah atau menolak izin objek dalam bucket. Kebijakan bucket mengizinkan atau menolak permintaan berdasarkan elemen dalam kebijakan. Elemen ini dapat mencakup pemohon, tindakan S3 pada Outposts, sumber daya, dan aspek atau ketentuan permintaan (misalnya, alamat IP yang digunakan untuk membuat permintaan). Misalnya, Anda dapat membuat kebijakan bucket yang memberikan izin lintas akun untuk mengunggah objek ke bucket S3 di Outposts sekaligus memastikan pemilik bucket memiliki kendali penuh atas objek yang diunggah. Untuk informasi selengkapnya, lihat Contoh kebijakan bucket Amazon S3.

Dalam kebijakan bucket, Anda dapat menggunakan karakter wildcard (`*`) di ARN dan nilai lainnya untuk memberikan izin ke subset objek. Misalnya, Anda dapat mengendalikan akses ke kelompok objek yang dimulai dengan prefiks umum atau diakhiri dengan ekstensi tertentu, seperti `.html`.

Titik akses S3 di Outposts

Titik akses S3 di Outposts diberi nama titik akhir jaringan dengan kebijakan akses khusus yang menjelaskan cara data dapat diakses menggunakan titik akhir tersebut. Titik akses menyederhanakan pengelolaan akses data dalam skala besar untuk set data bersama dalam S3 di Outposts. Titik akses dilampirkan ke bucket yang dapat Anda gunakan untuk melakukan operasi objek S3, seperti `GetObject` dan `PutObject`.

Saat menentukan bucket untuk operasi objek, Anda menggunakan ARN titik akses atau alias titik akses. Untuk informasi selengkapnya tentang alias titik akses, lihat Menggunakan alias untuk titik akses bucket S3 di Outposts Anda di bucket S3 di Outposts.

Titik akses memiliki izin dan kendali jaringan yang berbeda yang diterapkan S3 di Outposts untuk setiap permintaan yang dibuat melalui titik akses tersebut. Setiap titik akses memberlakukan kebijakan titik akses khusus yang bekerja bersama kebijakan bucket yang melekat pada bucket dasar.

Untuk informasi selengkapnya, lihat Mengakses S3 Anda di ember dan objek Outposts.

Fitur S3 di Outposts

Manajemen akses

S3 di Outposts menyediakan fitur untuk mengaudit dan mengelola akses ke bucket dan objek Anda. Secara default, S3 pada bucket Outposts dan objek di dalamnya bersifat pribadi. Anda hanya memiliki akses ke sumber daya S3 di Outposts yang Anda buat.

Untuk memberikan izin sumber daya terperinci yang mendukung kasus penggunaan spesifik Anda atau untuk mengaudit izin sumber daya S3 di Outposts, Anda dapat menggunakan fitur berikut.

- S3 Blokir Akses Publik–Memblokir akses publik ke bucket dan objek. Untuk bucket di Outposts, Blokir Akses Publik selalu diaktifkan secara default.
- AWS Identity and Access Management (IAM) — IAM adalah layanan web yang membantu Anda mengontrol akses ke AWS sumber daya dengan aman, termasuk sumber daya S3 on Outposts Anda. Dengan IAM, Anda dapat mengelola izin secara terpusat yang mengendalikan sumber daya AWS yang dapat diakses pengguna. Anda menggunakan IAM untuk mengontrol siapa yang diautentikasi (masuk) dan diotorisasi (memiliki izin) untuk menggunakan sumber daya.
- Titik akses S3 di Outposts–Mengelola akses data untuk set data bersama di S3 di Outposts. Titik akses diberi nama titik akhir jaringan dengan kebijakan akses khusus. Titik akses melekat ke bucket dan dapat digunakan untuk melakukan operasi objek, seperti `GetObject` dan `PutObject`.
- Kebijakan bucket–Gunakan bahasa kebijakan berbasis IAM untuk mengonfigurasi izin berbasis sumber daya untuk bucket S3 dan objek di dalamnya.
- AWS Resource Access Manager (AWS RAM) — Bagikan kapasitas S3 on Outposts Anda dengan aman di seluruh Akun AWS, di dalam organisasi atau unit organisasi (OU) Anda di. AWS Organizations

Pencatatan dan pemantauan penyimpanan

S3 di Outposts menyediakan alat bantu pencatatan dan pemantauan yang dapat Anda gunakan untuk memantau dan mengendalikan cara sumber daya S3 di Outposts Anda digunakan. Untuk informasi selengkapnya, lihat Alat pemantauan.

- CloudWatch Metrik Amazon untuk S3 di Outposts — Lacak kesehatan operasional sumber daya Anda dan pahami ketersediaan kapasitas Anda.

- CloudWatch Acara Amazon Events untuk S3 di Outposts - Buat aturan untuk setiap peristiwa S3 pada Outposts API untuk menerima pemberitahuan melalui CloudWatch semua target Acara yang didukung, termasuk Amazon Simple Queue Service (Amazon SQS), Amazon Simple Notification Service (Amazon SNS), dan. AWS Lambda
- AWS CloudTrail log untuk S3 di Outposts — Rekam tindakan yang diambil oleh pengguna, peran, atau Layanan AWS di S3 di Outposts. CloudTrail log memberi Anda pelacakan API terperinci untuk operasi tingkat ember dan tingkat objek S3.

Konsistensi kuat

S3 di Outposts memberikan konsistensi read-after-write yang kuat untuk permintaan PUT dan DELETE objek di bucket S3 on Outposts Anda secara keseluruhan. Wilayah AWS Perilaku ini berlaku untuk penulisan objek baru dan permintaan LETAKKAN yang menimpa objek yang ada serta untuk HAPUS permintaan. Selain itu, tag objek dan metadata objek S3 di Outposts (misalnya, objek HEAD) sangat konsisten. Untuk informasi selengkapnya, lihat Model konsistensi data Amazon S3.

Layanan terkait

Setelah Anda memuat data Anda ke S3 di Outposts, Anda dapat menggunakannya dengan Layanan AWS yang lain. Berikut adalah layanan yang mungkin paling sering Anda gunakan:

- Amazon Elastic Compute Cloud (Amazon EC2)–Menyediakan kapasitas komputasi yang aman dan dapat diskalakan di AWS Cloud. Menggunakan Amazon EC2 mengurangi kebutuhan Anda untuk berinvestasi pada perangkat keras di awal, sehingga Anda dapat mengembangkan dan mendeploy aplikasi lebih cepat. Anda dapat menggunakan Amazon EC2 untuk meluncurkan server virtual sebanyak atau sesedikit yang Anda butuhkan, mengonfigurasi keamanan dan jaringan, serta mengelola penyimpanan.
- Amazon Elastic Block Store (Amazon EBS) di Outposts–Gunakan snapshot lokal Amazon EBS pada Outposts untuk menyimpan snapshot volume pada Outpost secara lokal dalam S3 di Outposts.
- Amazon Relational Database Service (Amazon RDS) di Outposts–Gunakan cadangan lokal Amazon RDS untuk menyimpan cadangan Amazon RDS Anda secara lokal di Outpost Anda.
- AWS DataSync— Otomatiskan transfer data antara Outposts Anda dan Wilayah AWS, memilih apa yang akan ditransfer, kapan harus mentransfer, dan berapa banyak bandwidth jaringan yang akan digunakan. S3 di Outposts terintegrasi dengan. AWS DataSync Untuk aplikasi on-premise yang memerlukan pengolahan lokal dengan throughput tinggi, S3 di Outposts menyediakan

penyimpanan objek on-premise untuk meminimalkan transfer data dan buffer dari variasi jaringan, sekaligus memberi Anda kemampuan untuk mentransfer data antara Outposts dan Wilayah AWS dengan mudah.

Mengakses S3 di Outposts

Anda dapat bekerja dengan S3 di Outposts dengan salah satu cara berikut ini:

AWS Management Console

Konsol adalah antarmuka pengguna berbasis web untuk mengelola S3 di Outposts dan sumber daya AWS . Jika Anda telah mendaftar untuk Akun AWS, Anda dapat mengakses S3 di Outposts dengan masuk ke AWS Management Console dan memilih S3 dari AWS Management Console halaman beranda. Lalu pilih bucket Outposts dari panel navigasi kiri.

AWS Command Line Interface

Anda dapat menggunakan alat baris AWS perintah untuk mengeluarkan perintah atau membangun skrip di baris perintah sistem Anda untuk melakukan tugas AWS (termasuk S3).

The AWS Command Line Interface (AWS CLI) menyediakan perintah untuk satu set yang luas Layanan AWS. AWS CLI Ini didukung di Windows, macOS, dan Linux. Untuk memulai, lihat Panduan Pengguna AWS Command Line Interface. Untuk informasi selengkapnya tentang perintah yang dapat Anda gunakan dengan S3 di Outposts, lihat s3api, s3control, dan s3outposts dalam Referensi Perintah AWS CLI .

AWS SDK

AWS menyediakan SDK (perangkat pengembangan perangkat lunak) yang terdiri dari pustaka dan kode sampel untuk berbagai bahasa dan platform pemrograman (Java, Python, Ruby, .NET, iOS, Android, dan sebagainya). AWS SDK menyediakan cara mudah untuk membuat akses terprogram ke S3 di Outposts dan. AWS Karena S3 di Outposts menggunakan SDK yang sama dengan Amazon S3, S3 di Outposts memberikan pengalaman yang konsisten menggunakan API, otomatisasi, dan alat bantu S3 yang sama.

S3 di Outposts adalah layanan REST. Anda dapat mengirim permintaan ke S3 di Outposts menggunakan pustaka SDK AWS , yang membungkus API REST yang mendasarinya dan menyederhanakan tugas pemrograman Anda. Misalnya, SDK menangani tugas seperti menghitung tanda tangan, menandatangani permintaan secara kriptografis, mengelola kesalahan, dan mencoba

kembali permintaan secara otomatis. Untuk informasi tentang AWS SDK, termasuk cara mengunduh dan menginstalnya, lihat Alat untuk Dibangun AWS.

Membayar untuk S3 di Outposts

Anda dapat membeli berbagai konfigurasi AWS Outposts rak yang menampilkan kombinasi jenis instans Amazon EC2, volume gp2 solid state drive (SSD) Amazon EBS General Purpose (SSD), dan S3 di Outposts. Harga termasuk pengiriman, instalasi, pemeliharaan layanan infrastruktur, serta patch dan peningkatan perangkat lunak.

Untuk informasi selengkapnya lihat AWS Outposts harga rak.

Langkah selanjutnya

Untuk informasi selengkapnya tentang bekerja dengan S3 di Outposts, lihat topik berikut:

- Menyiapkan Anda
- Bagaimana Amazon S3 di Outposts berbeda dari Amazon S3?
- Memulai dengan Amazon S3 di Outposts
- Jaringan untuk S3 di Outposts
- Bekerja dengan S3 di bucket Outposts
- Bekerja dengan S3 di objek Outposts
- Keamanan di S3 di Outposts
- Mengelola penyimpanan S3 di Outposts
- Mengembangkan dengan Amazon S3 di Outposts

Menyiapkan Anda

Untuk memulai dengan Amazon S3 di Outposts, Anda memerlukan Outpost dengan kapasitas Amazon S3 yang diterapkan di fasilitas Anda. Untuk informasi tentang opsi untuk pemesanan kapasitas Outdan Outpost dan S3, lihat AWS Outposts. Untuk memeriksa apakah Outposts Anda memiliki kapasitas S3 di atasnya, Anda dapat menggunakan panggilan API ListOutpostsWithS3. Untuk spesifikasi dan untuk melihat bagaimana S3 di Outposts berbeda dari Amazon S3, lihatBagaimana Amazon S3 di Outposts berbeda dari Amazon S3?

Untuk informasi lain, lihat topik berikut.

Topik

- Pesan Pos Terdepan baru

Pesan Pos Terdepan baru

Jika Anda perlu memesan Outpost baru dengan kapasitas S3, lihat HargaAWS Outposts rak untuk memahami pilihan kapasitas untuk Amazon Elastic Compute Cloud (Amazon EC2), Amazon Elastic Block Store (Amazon EBS), dan Amazon S3.

Setelah memilih konfigurasi Anda, ikuti langkah dalam Buat Outpost dan pesan kapasitas Outdi PanduanAWS Outposts Pengguna.

Bagaimana Amazon S3 di Outposts berbeda dari Amazon S3?

Amazon S3 on Outposts mengirimkan penyimpanan objek keAWS Outposts lingkungan lokal Anda. Menggunakan S3 di Outposts membantu Anda memenuhi pemrosesan lokal, residensi data, dan kebutuhan kinerja yang menuntut dengan menjaga data tetap dekat dengan aplikasi lokal. Karena menggunakan API dan fitur Amazon S3, S3 on Outposts memudahkan untuk menyimpan, mengamankan, menandai, melaporkan, dan mengontrol akses ke data di Outposts Anda dan memperluasAWS infrastruktur ke fasilitas lokal Anda untuk pengalaman hibrida yang konsisten.

Untuk informasi lebih lanjut tentang bagaimana S3 di Outposts adalah unik, lihat topik berikut.

Topik

- S3 di spesifikasi Outposts
- Operasi API yang didukung oleh S3 di Outposts
- Fitur Amazon S3 tidak didukung oleh S3 di Outposts
- S3 di persyaratan jaringan Outposts

S3 di spesifikasi Outposts

- Ukuran maksimum bucket Outposts adalah 50 TB.
- Jumlah maksimum bucket Outposts adalah 100 perAkun AWS.
- Bucket Outposts hanya dapat diakses dengan menggunakan titik akses dan titik akhir.
- Jumlah maksimum titik akses per bucket Outposts adalah 10.

- Kebijakan titik akses dibatasi sebesar 20 KB ukurannya.
- Pemilik Outpost dapat mengelola akses dalam organisasi AndaAWS Organizations dengan menggunakanAWS Resource Access Manager. Semua akun yang membutuhkan akses ke Pos Terdepan harus berada dalam organisasi yang sama dengan akun pemilik diAWS Organizations.
- S3 pada akun pemilik bucket Outposts selalu menjadi pemilik semua objek di dalam bucket.
- Hanya S3 pada akun pemilik bucket Outposts yang dapat melakukan operasi pada bucket.
- Batasan ukuran objek sesuai dengan Amazon S3.
- Semua objek yang disimpan di S3 di Outposts disimpan di kelas penyimpanan `OUTPOSTS`.
- Secara default, semua objek yang disimpan di kelas`OUTPOSTS` penyimpanan disimpan dengan enkripsi sisi server dengan kunci enkripsi terkelola Amazon S3 (SSE-S3). Anda juga dapat secara eksplisit memilih untuk menyimpan objek dengan enkripsi sisi server dengan kunci enkripsi yang disediakan pelanggan (SSE-C).
- Jika tidak tersedia cukup ruang untuk menyimpan objek di Outpost Anda, API mengembalikan pengecualian kapasitas yang tidak memadai (ICE).

Operasi API yang didukung oleh S3 di Outposts

Untuk daftar operasi API yang didukung oleh S3 di Outposts, lihatAmazon S3 di operasi API Outposts.

Fitur Amazon S3 tidak didukung oleh S3 di Outposts

Fitur Amazon S3 saat ini tidak didukung oleh Amazon S3 di Outposts. Upaya apa pun untuk menggunakannya ditolak.

- Daftar kontrol akses (ACL)
- Cross-origin resource sharing (CORS)
- Operasi Batch S3
- Laporan Inventaris S3
- Mengubah enkripsi bucket default
- Bucket publik
- Multi-factor authentication (MFA) Delete
- Transisi S3 Siklus hidup (selain dari penghapusan objek dan pembatalan unggahan multibagian yang tidak lengkap)

- Pegangan hukum S3 Object Lock
- Retensi Object Lock
- Enkripsi Sisi server dengan kunciAWS Key Management Service (AWS KMS) (SSE-KMS)
- S3 Replication Time Control (S3 RTC)
- CloudWatch Metrik permintaan Amazon
- Konfigurasi metrik
- Transfer Acceleration
- Notifikasi Peristiwa S3
- Bucket Pemohon Membayar
- S3 Select
- Peristiwa AWS Lambda
- Pencatatan akses server
- Permintaan HTTP POST
- SOAP
- Akses situs web

S3 di persyaratan jaringan Outposts

- Untuk perutean permintaan ke S3 di titik akses Outposts, Anda harus membuat dan mengonfigurasi S3 di titik akhir Outposts. Batasan berikut ini berlaku ke titik akhir untuk S3 di Outposts:
  - Setiap virtual private cloud (VPC) pada Outposts dapat memiliki satu titik akhir terkait, dan Anda dapat memiliki hingga 100 titik akhir per Outposts.
  - Anda dapat memetakan beberapa titik akses ke titik akhir yang sama.
  - Anda dapat menambahkan titik akhir hanya ke VPC dengan blok CIDR di subruang dari rentang CIDR berikut:
    - 10.0.0.0/8
    - 172.16.0.0/12
    - 192.168.0.0/16
- Anda dapat membuat titik akhir ke Outposts hanya dari VPC yang memiliki blok CIDR yang tidak saling timpa.
- Anda dapat membuat titik akhir hanya dari dalam subnet Outpostsnya.

- Subnet yang Anda gunakan untuk membuat titik akhir harus berisi empat alamat IP untuk S3 di Outposts untuk digunakan.
- Jika Anda menentukan kumpulan alamat IP milik pelanggan (CoIP pool), itu harus berisi empat alamat IP untuk S3 di Outposts untuk digunakan.
- Anda hanya dapat membuat satu endpoint per Outpost per VPC.

Memulai dengan Amazon S3 di Outposts

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di Outposts AWS Anda dan dengan mudah menyimpan dan mengambil objek on-premise untuk aplikasi yang memerlukan akses data lokal, pengolahan data lokal, dan residensi data. S3 di Outposts menyediakan kelas penyimpanan baru, S3 Outposts (`OUTPOSTS`), yang menggunakan API Amazon S3, dan dirancang untuk menyimpan data secara tahan lama dan redundan di beberapa perangkat dan server diAWS Outposts. Anda berkomunikasi dengan bucket Outpost dengan menggunakan titik akses dan koneksi titik akhir melalui cloud pribadi virtual (VPC). Anda dapat menggunakan API dan fitur yang sama di bucket Outpost seperti yang Anda lakukan di bucket Amazon S3, seperti kebijakan akses, enkripsi, dan pemberian tag. Anda dapat menggunakan S3 on Outposts melaluiAWS Management Console,AWS Command Line Interface(AWS CLI),AWSSDK, atau REST API.

Dengan Amazon S3 di Outpost, Anda dapat menggunakan API dan fitur Amazon S3, seperti penyimpanan objek, kebijakan akses, enkripsi, dan pemberian tag, di AWS Outposts seperti yang Anda lakukan di Amazon S3. Untuk informasi tentang S3 di Outposts, lihatApa itu Amazon S3 di Outposts?

Topik

- Mengatur IAM dengan S3 di Outposts
- Memulai dengan menggunakanAWS Management Console
- Memulai dengan menggunakan AWS CLI dan SDK for Java

Mengatur IAM dengan S3 di Outposts

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengendalikan siapa saja yang dapat diautentikasi (masuk) dan diizinkan (memiliki izin) untuk menggunakan sumber daya Amazon S3 di Outposts. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan. Secara default, pengguna tidak memiliki izin untuk sumber daya dan operasi S3 di

Outposts. Untuk memberikan izin akses bagi sumber daya dan operasi API S3 di Outposts, Anda dapat menggunakan IAM untuk membuat pengguna, grup, atau peran dan melampirkan izin.

Untuk memberikan akses, menambahkan izin ke pengguna, grup, atau peran Anda:

- Pengguna dan grup di AWS IAM Identity Center:

  Buat rangkaian izin. Ikuti instruksi di Buat rangkaian izin di Panduan Pengguna AWS IAM Identity Center .
- Pengguna yang dikelola di IAM melalui penyedia identitas:

  Buat peran untuk federasi identitas. Ikuti instruksi dalam Membuat peran untuk penyedia identitas pihak ketiga (federasi) di Panduan Pengguna IAM.
- Pengguna IAM:
  - Buat peran yang dapat diambil pengguna Anda. Ikuti instruksi dalam Membuat peran untuk pengguna IAM dalam Panduan Pengguna IAM.
  - (Tidak disarankan) Pasang kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti instruksi dalam Menambahkan izin ke pengguna (konsol) dalam Panduan Pengguna IAM.

Selain kebijakan berbasis identitas IAM, S3 di Outposts mendukung kebijakan bucket dan titik akses. Kebijakan bucket dan titik akses adalah kebijakan berbasis sumber daya yang dilampirkan ke sumber daya S3 di Outposts.

- Kebijakan bucket dilampirkan pada bucket dan mengizinkan atau menolak permintaan ke bucket serta objek di dalamnya berdasarkan elemen dalam kebijakan.
- Sebaliknya, kebijakan titik akses dilampirkan ke titik akses dan memungkinkan atau menolak permintaan ke titik akses.

Kebijakan titik akses bekerja dengan kebijakan bucket yang dilampirkan pada bucket yang mendasari S3 di Outposts. Agar aplikasi atau pengguna dapat mengakses objek di bucket S3 di Outposts melalui titik akses S3 di Outposts, kebijakan titik akses dan kebijakan bucket harus mengizinkan permintaan tersebut.

Pembatasan yang Anda sertakan dalam kebijakan titik akses hanya berlaku untuk permintaan yang dibuat melalui titik akses tersebut. Misalnya, jika titik akses dilampirkan ke bucket, Anda tidak dapat menggunakan kebijakan titik akses untuk mengizinkan atau menolak permintaan yang

dibuat langsung ke bucket. Namun, pembatasan yang Anda terapkan pada kebijakan bucket dapat mengizinkan atau menolak permintaan yang dibuat langsung ke bucket atau melalui titik akses.

Dalam kebijakan IAM atau kebijakan berbasis sumber daya, Anda menentukan tindakan S3 di Outposts mana yang diizinkan atau ditolak. Tindakan S3 di Outposts sesuai dengan operasi API S3 di Outposts tertentu. Tindakan S3 di Outposts menggunakan prefiks namespace `s3-outposts:`. Permintaan yang dibuat ke S3 on Outposts mengontrol API dalam dan permintaan Wilayah AWS yang dibuat ke titik akhir API objek di Outpost diautentikasi dengan menggunakan IAM dan diotorisasi terhadap awalan namespace. `s3-outposts:` Untuk bekerja dengan S3 di Outposts, konfigurasikan pengguna IAM Anda dan izinkan mereka terhadap namespace IAM `s3-outposts:`.

Untuk informasi lebih lanjut, lihat Tindakan, sumber daya, dan kunci syarat untuk Amazon S3 di Outposts di Referensi Otorisasi Layanan.

> **Note**
>
> - Daftar kontrol akses (ACL) tidak didukung oleh S3 di Outposts.
> - S3 di Outposts adalah default bagi pemilik bucket sebagai pemilik objek, untuk membantu memastikan bahwa pemilik bucket tidak dapat dicegah untuk mengakses atau menghapus objek.
> - S3 di Outposts selalu mengaktifkan Blokir Akses Publik S3 untuk membantu memastikan objek tidak pernah memiliki akses publik.

Untuk informasi tentang pengaturan IAM untuk S3 di Outposts, lihat topik berikut.

Topik

- Pengguna utama kebijakan S3 di Outposts
- ARN sumber daya untuk S3 di Outposts
- Contoh kebijakan untuk S3 di Outposts
- Izin untuk titik akhir S3 di Outposts
- Peran yang ditautkan dengan layanan untuk S3 di Outposts

Pengguna utama kebijakan S3 di Outposts

Saat membuat kebijakan berbasis sumber daya untuk memberikan akses ke bucket S3 di Outposts, Anda harus menggunakan elemen `Principal` tersebut untuk menentukan orang atau aplikasi yang dapat membuat permintaan tindakan atau operasi pada sumber daya tersebut. Untuk kebijakan S3 di Outposts, Anda dapat menggunakan salah satu pengguna utama berikut:

- Sebuah Akun AWS
- Pengguna IAM
- Peran IAM
- Semua pengguna utama, dengan menentukan karakter wildcard (*) dalam kebijakan yang menggunakan elemen `Condition` untuk membatasi akses ke rentang IP tertentu

> **Important**
>
> Anda tidak dapat menulis kebijakan untuk bucket S3 di Outposts yang menggunakan karakter wildcard (`*`) dalam elemen `Principal` kecuali jika kebijakan tersebut juga menyertakan `Condition` yang membatasi akses ke rentang alamat IP tertentu. Pembatasan ini membantu memastikan tidak ada akses publik ke bucket S3 di Outposts Anda. Sebagai contoh, lihat Contoh kebijakan untuk S3 di Outposts.

Untuk informasi selengkapnya tentang elemen `Principal`, lihat elemen kebijakan JSON AWS : Pengguna utama dalam Panduan Pengguna IAM.

ARN sumber daya untuk S3 di Outposts

Amazon Resource Names (ARN) untuk S3 di Outposts berisi Outpost ID selain tempat Outpost berada, ID, dan nama resource. Wilayah AWS Akun AWS Untuk mengakses dan melakukan tindakan pada bucket dan objek Outposts Anda, Anda harus menggunakan salah satu format ARN yang ditunjukkan pada tabel berikut.

*`partition`*Nilai dalam ARN mengacu pada sekelompok. Wilayah AWS Masing-masing Akun AWS dicakup ke satu partisi. Berikut ini adalah partisi yang didukung:

- `aws` – Wilayah AWS
- `aws-us-gov`— AWS GovCloud (US) Daerah

Format ARN S3 di Outposts

ARN Amazon S3 di Outposts	Format ARN	Contoh
ARN Bucket	`arn:partition :s3-outposts: region: account_id :outpost / outpost_id / bucket/bucket_name`	`arn:aws:s3-outposts: us-west-2 :123456789012 : outpost/ op-01ac5d28a6a232904 / bucket/example-s3-bucket1`
Titik Akses ARN	`arn:partition :s3-outposts: region: account_id :outpost / outpost_id /accesspoint/ accesspoint_name`	`arn:aws:s3-outposts: us-west-2 :123456789012 : outpost/ op-01ac5d28a6a232904 /accesspoint/ access-point-name`
ARN objek	`arn:partition :s3-outposts: region: account_id :outpost / outpost_id / bucket/bucket_name / object/object_key`	`arn:aws:s3-outposts: us-west-2 :123456789012 : outpost/ op-01ac5d28a6a232904 / bucket/example-s3-bucket1 /object/myobject`
ARN objek titik akses S3 di Outposts (digunakan dalam kebijakan)	`arn:partition :s3-outposts: region: account_id :outpost / outpost_id /accesspoint/ accesspoint_name / object/object_key`	`arn:aws:s3-outposts: us-west-2 :123456789012 : outpost/ op-01ac5d28a6a232904 /accesspoint/ access-point-name/object/myobject`

ARN Amazon S3 di Outposts	Format ARN	Contoh
ARN S3 di Outposts	`arn:partition :s3-outposts: region: account_id :outpost / outpost_id`	`arn:aws:s3-outposts: us-west-2 :123456789012 : outpost/ op-01ac5d28a6a232904`

Contoh kebijakan untuk S3 di Outposts

Example : S3 tentang kebijakan bucket Outposts dengan kepala sekolah Akun AWS

Kebijakan bucket berikut menggunakan Akun AWS prinsipal untuk memberikan akses ke bucket S3 di Outposts. Untuk menggunakan kebijakan bucket ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
{
   "Version":"2012-10-17",
   "Id":"ExampleBucketPolicy1",
   "Statement":[
      {
         "Sid":"statement1",
         "Effect":"Allow",
         "Principal":{
            "AWS":"123456789012"
         },
         "Action":"s3-outposts:*",
         "Resource":"arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket"
      }
   ]
}
```

Example : kebijakan bucket S3 pada Outposts dengan pengguna utama wildcard (**`*`**) dan kunci syarat untuk membatasi akses ke rentang alamat IP tertentu

Kebijakan bucket berikut menggunakan pengguna utama wildcard (`*`) dengan syarat `aws:SourceIp` untuk membatasi akses ke rentang alamat IP tertentu. Untuk menggunakan kebijakan bucket ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
{
    "Version": "2012-10-17",
    "Id": "ExampleBucketPolicy2",
    "Statement": [
        {
            "Sid": "statement1",
            "Effect": "Allow",
            "Principal": { "AWS" : "*" },
            "Action":"s3-outposts:*",
            "Resource":"arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket",
            "Condition" : {
                "IpAddress" : {
                    "aws:SourceIp": "192.0.2.0/24"
                },
                "NotIpAddress" : {
                    "aws:SourceIp": "198.51.100.0/24"
                }
            }
        }
    ]
}
```

Izin untuk titik akhir S3 di Outposts

S3 di Outposts memerlukan izinnya sendiri di IAM untuk mengelola tindakan titik akhir S3 di Outposts.

> **Note**
>
> - Untuk titik akhir yang menggunakan jenis akses kumpulan alamat IP (kumpulan CoIP) milik pelanggan, Anda juga harus memiliki izin untuk bekerja dengan alamat IP dari kumpulan CoIP Anda, seperti yang dijelaskan dalam tabel berikut.
> - Untuk akun bersama yang mengakses S3 di Outposts dengan AWS Resource Access Manager menggunakan, pengguna di akun bersama ini tidak dapat membuat titik akhir mereka sendiri di subnet bersama. Apabila pengguna di akun bersama ingin mengelola titik akhir mereka sendiri, akun bersama harus membuat subnetnya sendiri di Outpost. Untuk informasi selengkapnya, lihat the section called “Berbagi S3 di Outposts”.

Izin IAM terkait titik akhir S3 di Outposts

Tindakan	Izin IAM

`CreateEndpoint`	`s3-outposts:CreateEndpoint`
	
	`ec2:CreateNetworkInterface`
	
	`ec2:DescribeNetworkInterfaces`
	
	`ec2:DescribeVpcs`
	
	`ec2:DescribeSecurityGroups`
	
	`ec2:DescribeSubnets`
	
	`ec2:CreateTags`
	
	`iam:CreateServiceLinkedRole`
	
	Untuk titik akhir yang menggunakan jenis akses kumpulan alamat IP (kumpulan CoIP) milik pelanggan on-premise, izin tambahan berikut diperlukan:
	
	`s3-outposts:CreateEndpoint`
	
	`ec2:DescribeCoipPools`
	
	`ec2:GetCoipPoolUsage`
	
	`ec2:AllocateAddress`
	
	`ec2:AssociateAddress`
	
	`ec2:DescribeAddresses`
	
	`ec2:DescribeLocalGatewayRouteTableVpcAssociations`

`DeleteEndpoint`	`s3-outposts:DeleteEndpoint`
	
	`ec2:DeleteNetworkInterface`

Tindakan	Izin IAM

	`ec2:DescribeNetworkInterfaces`
	
	Untuk titik akhir yang menggunakan jenis akses kumpulan alamat IP (kumpulan CoIP) milik pelanggan on-premise, izin tambahan berikut diperlukan:
	
	`s3-outposts:DeleteEndpoint`
	
	`ec2:DisassociateAddress`
	
	`ec2:DescribeAddresses`
	
	`ec2:ReleaseAddress`

`ListEndpoints`	`s3-outposts:ListEndpoints`

> **Note**
>
> Anda dapat menggunakan tag sumber daya dalam kebijakan IAM untuk mengelola izin.

Peran yang ditautkan dengan layanan untuk S3 di Outposts

S3 di Outposts menggunakan peran yang ditautkan dengan layanan IAM untuk membuat beberapa sumber daya jaringan atas nama Anda. Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan untuk Amazon S3 di Outposts.

Memulai dengan menggunakanAWS Management Console

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di Outposts AWS Anda dan dengan mudah menyimpan dan mengambil objek on-premise untuk aplikasi yang memerlukan akses data lokal, pengolahan data lokal, dan residensi data. S3 pada Outposts menyediakan kelas penyimpanan baru, S3 Outposts (`OUTPOSTS`), yang menggunakan API Amazon S3, dan dirancang untuk menyimpan data secara tahan lama dan redundan di beberapa perangkat dan server di AndaAWS Outposts. Anda berkomunikasi dengan bucket Outpost Anda dengan menggunakan titik akses dan koneksi titik akhir melalui cloud pribadi virtual (VPC). Anda dapat menggunakan API dan

fitur yang sama di bucket Outpost seperti yang Anda lakukan di bucket Amazon S3, seperti kebijakan akses, enkripsi, dan pemberian tag. Anda dapat menggunakan S3 pada Outposts melaluiAWS Management Console,AWS Command Line Interface (AWS CLI),AWS SDK, atau REST API. Untuk informasi selengkapnya, lihat Apa itu Amazon S3 di Outposts?

Untuk memulai dengan S3 pada Outposts dengan menggunakan konsol, lihat topik berikut. Untuk memulai dengan menggunakanAWS CLI atauAWS SDK for Java, lihatMemulai dengan menggunakan AWS CLI dan SDK for Java.

Topik

- Buat bucket, titik akses, dan titik akhir
- Langkah selanjutnya

Buat bucket, titik akses, dan titik akhir

Prosedur berikut ini menunjukkan cara membuat bucket pertama Anda di S3 pada Outposts. Saat Anda membuat bucket menggunakan konsol, Anda juga membuat titik akses dan titik akhir yang terkait dengan bucket sehingga Anda dapat segera mulai menyimpan objek di bucket Anda.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih bucket Outposts.
3. Pilih Buat bucket Outposts.
4. Untuk nama bucket, masukkan nama yang sesuai dengan Sistem Nama Domain (DNS) untuk bucket Anda.

   Nama kelompok harus:

   - Jadilah unik dalamAkun AWS, Outpost, danWilayah AWS bahwa Outpost adalah homed ke.
   - Panjangnya 3—63 karakter.
   - Tidak mengandung karakter huruf besar.
   - Mulai dengan huruf atau angka kecil.

   Setelah membuat bucket, Anda tidak dapat mengubah namanya. Untuk informasi tentang penamaan bucket, lihat Peraturan penamaan bucket.

> **Important**
>
> Hindari menyertakan informasi sensitif seperti nomor rekening dalam nama bucket. Nama bucket terlihat di URL yang mengarah ke objek di bucket.

5. Untuk Outpost, pilih Outpost tempat Anda ingin bucket berada.
6. Di bawah Bucket Versioning, atur status Versi S3 untuk bucket S3 on Outposts ke salah satu opsi berikut:

   - Nonaktifkan (default) - Bucket tetap tidak berversi.
   - Aktifkan - Memungkinkan Versi S3 untuk objek di bucket. Semua objek yang ditambahkan ke bucket menerima ID versi unik.

   Untuk informasi selengkapnya tentang S3 Versioning, lihat Mengelola versioning.
7. (Opsional) Tambahkan tag opsional yang ingin Anda kaitkan dengan bucket Outposts. Anda dapat menggunakan tag untuk melacak kriteria untuk proyek individu atau grup proyek, atau untuk memberi label bucket Anda dengan menggunakan tag alokasi biaya.

   Secara default, semua objek yang disimpan di bucket Outposts Anda disimpan menggunakan enkripsi sisi server dengan kunci enkripsi terkelola Amazon S3 (SSE-S3). Anda juga dapat secara eksplisit memilih untuk menyimpan objek dengan enkripsi yang disediakan pelanggan (SSE-C). Untuk mengubah jenis enkripsi, Anda harus menggunakan REST API,AWS Command Line Interface (AWS CLI), atauAWS SDK.
8. Di bagian Pengaturan titik akses Outposts, masukkan nama titik akses.

   Titik akses S3 di Outposts menyederhanakan pengelolaan akses data dalam skala besar untuk kumpulan data bersama di S3 di Outposts. Titik akses diberi nama titik akhir jaringan yang dilampirkan ke bucket Outposts yang dapat Anda gunakan untuk melakukan operasi objek S3. Untuk informasi selengkapnya, lihat Titik Akses.

   Nama titik akses harus unik dalam akun untuk Wilayah dan Outposts ini, dan patuh dengan Pembatasan dan batasan titik akses.
9. Pilih VPC untuk titik akses Amazon S3 di Outposts ini.

   Jika Anda tidak memiliki VPC, pilih Dengan VPC. Untuk informasi selengkapnya, lihat Membuat titik akses terbatas pada cloud privat virtual.

Virtual private cloud (VPC) memungkinkan Anda meluncurkan sumber daya AWS ke jaringan virtual yang Anda tentukan. Jaringan virtual ini sangat mirip dengan jaringan tradisional yang akan Anda operasikan di pusat data Anda sendiri, dengan manfaat dari penggunaan infrastruktur AWS yang dapat diskalakan.

10. (Opsional untuk VPC yang sudah ada) Pilih subnet Endpoint untuk endpoint Anda.

    Subnet adalah serangkaian alamat IP di VPC Anda. Jika Anda tidak memiliki subnet yang Anda inginkan, pilih Buat subnet. Untuk informasi selengkapnya, lihat Jaringan untuk S3 di Outposts.

11. (Opsional untuk VPC yang sudah ada) Pilih grup keamanan Endpoint untuk endpoint Anda.

    Grup keamanan bertindak sebagai firewall virtual untuk mengontrol lalu lintas masuk dan keluar.

12. (Opsional untuk VPC yang ada) Pilih jenis akses Endpoint:

    - Pribadi - Untuk digunakan dengan VPC.
    - IP milik pelanggan - Untuk digunakan dengan Kumpulan alamat IP milik pelanggan (kumpulan CoIP) dari dalam jaringan lokal Anda.

13. (Opsional) Tentukan kebijakan titik akses. Konsol secara otomatis menampilkan Amazon Resource Name (ARN) untuk titik akses, yang dapat Anda gunakan dalam kebijakan.
14. Pilih Buat bucket Outposts.

> **Note**
>
> Diperlukan waktu hingga 5 menit agar titik akhir Outpost Anda untuk dibuat dan bucket Anda siap digunakan. Untuk mengkonfigurasi pengaturan bucket tambahan, pilih Melihat detail.

Langkah selanjutnya

Dengan Amazon S3 di Outposts, data objek selalu disimpan di Outpost. SaatAWS memasang rak Outpost, data Anda tetap lokal ke Outpost Anda untuk memenuhi persyaratan data-residensi. Objek Anda tidak pernah meninggalkan Outpost Anda dan tidak dalamWilayah AWS. Karena di-host In-region, Anda tidak dapat menggunakan konsol untuk mengunggah atau mengelola objek di Outpost Anda.AWS Management Console Namun, Anda dapat menggunakan REST API,AWS Command Line Interface (AWS CLI), danAWS SDK untuk mengunggah dan mengelola objek Anda melalui titik akses Anda.

Setelah membuat bucket, titik akses, dan titik akhir S3 di Outposts, Anda dapat menggunakan SDKAWS CLI atau Java untuk mengunggah objek ke bucket Anda. Untuk informasi selengkapnya, lihat Unggah objek ke ember S3 di Outposts.

Memulai dengan menggunakan AWS CLI dan SDK for Java

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di AWS Outposts dan dengan mudah menyimpan dan mengambil objek di tempat untuk aplikasi yang memerlukan akses data lokal, pemrosesan data lokal, dan residensi data. S3 on Outposts menyediakan kelas penyimpanan baru, S3 Outposts `OUTPOSTS` (), yang menggunakan API Amazon S3, dan dirancang untuk menyimpan data secara tahan lama dan berlebihan di beberapa perangkat dan server di perangkat Anda. AWS Outposts Anda berkomunikasi dengan bucket Outposts menggunakan titik akses dan koneksi titik akhir melalui cloud privat virtual (VPC). Anda dapat menggunakan API dan fitur yang sama di bucket Outposts seperti yang Anda lakukan di bucket Amazon S3, termasuk kebijakan akses, enkripsi, dan pemberian tag. Anda dapat menggunakan S3 di Outposts melalui AWS Management Console AWS Command Line Interface ,AWS CLI(), SDK AWS , atau REST API. Untuk informasi selengkapnya, lihat Apa itu Amazon S3 di Outposts?

Untuk memulai S3 di Outposts, Anda harus membuat bucket, titik akses, dan titik akhir. Kemudian, Anda dapat mengunggah objek ke bucket Anda. Contoh berikut menunjukkan kepada Anda bagaimana memulai dengan S3 di Outposts dengan menggunakan AWS CLI dan SDK for Java. Untuk memulai menggunakan konsol, lihat Memulai dengan menggunakanAWS Management Console.

Topik

- Langkah 1: Buat bucket
- Langkah 2: Buat titik akses
- Langkah 3: Buat titik akhir
- Langkah 4: Unggah objek ke bucket S3 di Outposts

Langkah 1: Buat bucket

Contoh berikut AWS CLI dan SDK for Java menunjukkan cara membuat bucket S3 di Outposts.

AWS CLI

Example

Contoh berikut membuat bucket S3 di Outposts (`s3-outposts:CreateBucket`) menggunakan AWS CLI. Untuk menjalankan perintah ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control create-bucket --bucket example-outposts-bucket --outpost-
id op-01ac5d28a6a232904
```

SDK for Java

Example

Contoh berikut membuat bucket S3 di Outposts (`s3-outposts:CreateBucket`) menggunakan SDK for Java.

```
import com.amazonaws.services.s3control.model.*;

public String createBucket(String bucketName) {

    CreateBucketRequest reqCreateBucket = new CreateBucketRequest()
            .withBucket(bucketName)
            .withOutpostId(OutpostId)
            .withCreateBucketConfiguration(new CreateBucketConfiguration());

    CreateBucketResult respCreateBucket =
 s3ControlClient.createBucket(reqCreateBucket);
    System.out.printf("CreateBucket Response: %s%n", respCreateBucket.toString());

    return respCreateBucket.getBucketArn();

}
```

Langkah 2: Buat titik akses

Untuk mengakses bucket Amazon S3 di Outposts, Anda harus membuat dan mengonfigurasi titik akses. Ini contoh bagaimana Anda cara membuat titik akses dengan menggunakan AWS CLI dan SDK for Java.

Titik akses menyederhanakan pengelolaan akses data dalam skala besar untuk set data bersama di Amazon S3. Titik akses diberi nama titik akhir jaringan yang melekat ke bucket yang dapat Anda gunakan untuk melakukan operasi objek Amazon S3, seperti `GetObject` dan `PutObject`. Dengan S3 di Outposts, Anda harus menggunakan titik akses untuk mengakses objek apa pun dalam bucket Outposts. Titik akses hanya mendukung virtual-host-style pengalamatan.

AWS CLI

Example

AWS CLI Contoh berikut membuat titik akses untuk bucket Outposts. Untuk menjalankan perintah ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control create-access-point --account-id 123456789012
 --name example-outposts-access-point --bucket "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket" --vpc-configuration VpcId=example-vpc-12345
```

SDK for Java

Example

Contoh SDK for Java berikut membuat titik akses untuk bucket Outposts. Untuk menggunakan contoh ini, ganti *`user input placeholders`* dengan informasi Anda sendiri.

```
import com.amazonaws.services.s3control.model.*;

public String createAccessPoint(String bucketArn, String accessPointName) {

    CreateAccessPointRequest reqCreateAP = new CreateAccessPointRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn)
            .withName(accessPointName)
            .withVpcConfiguration(new VpcConfiguration().withVpcId("vpc-12345"));

    CreateAccessPointResult respCreateAP =
 s3ControlClient.createAccessPoint(reqCreateAP);
    System.out.printf("CreateAccessPoint Response: %s%n", respCreateAP.toString());

    return respCreateAP.getAccessPointArn();

}
```

Langkah 3: Buat titik akhir

Untuk merutekan permintaan ke titik akses Amazon S3 di Outposts, Anda harus membuat dan mengonfigurasikan titik akhir S3 di Outposts. Untuk membuat titik akhir, Anda akan memerlukan koneksi aktif dengan tautan layanan ke wilayah asal Outposts Anda. Setiap Cloud Privat Virtual (VPC) pada Outposts Anda dapat memiliki satu titik akhir terkait. Untuk informasi selengkapnya tentang kuota titik akhir, lihat S3 di persyaratan jaringan Outposts. Anda harus membuat titik akhir agar dapat mengakses bucket Outposts Anda dan melakukan operasi objek. Untuk informasi selengkapnya, lihat Titik akhir.

Contoh-contoh ini menunjukkan cara membuat endpoint dengan menggunakan AWS CLI dan SDK for Java. Untuk informasi selengkapnya tentang izin yang diperlukan untuk membuat dan mengelola titik akhir, lihat Izin untuk titik akhir S3 di Outposts.

AWS CLI

Example

AWS CLI Contoh berikut membuat endpoint untuk Outpost dengan menggunakan jenis akses sumber daya VPC. VPC berasal dari subnet. Untuk menjalankan perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1
```

AWS CLI Contoh berikut membuat endpoint untuk Outpost dengan menggunakan jenis akses pool alamat IP (CoIP pool) milik pelanggan. Untuk menjalankan perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1 --access-type CustomerOwnedIp --
customer-owned-ipv4-pool ipv4pool-coip-12345678901234567
```

SDK for Java

Example

Contoh SDK untuk Java berikut membuat titik akhir untuk Outpost. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
import com.amazonaws.services.s3outposts.model.CreateEndpointRequest;
import com.amazonaws.services.s3outposts.model.CreateEndpointResult;

public void createEndpoint() {
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    CreateEndpointRequest createEndpointRequest = new CreateEndpointRequest()
                .withOutpostId("op-0d79779cef3c30a40")
                .withSubnetId("subnet-8c7a57c5")
                .withSecurityGroupId("sg-ab19e0d1")
                .withAccessType("CustomerOwnedIp")
                .withCustomerOwnedIpv4Pool("ipv4pool-coip-12345678901234567");
    // Use .withAccessType and .withCustomerOwnedIpv4Pool only when the access type
 is
    // customer-owned IP address pool (CoIP pool)
    CreateEndpointResult createEndpointResult =
 s3OutpostsClient.createEndpoint(createEndpointRequest);
    System.out.println("Endpoint is created and its ARN is " +
 createEndpointResult.getEndpointArn());
}
```

Langkah 4: Unggah objek ke bucket S3 di Outposts

Untuk mengunggah objek, lihatUnggah objek ke ember S3 di Outposts.

Jaringan untuk S3 di Outposts

Anda dapat menggunakan Amazon S3 di Outposts untuk menyimpan dan mengambil objek lokal untuk aplikasi yang memerlukan akses data lokal, pemrosesan data, dan residensi data. Bagian ini menjelaskan persyaratan jaringan untuk mengakses S3 di Outposts.

Topik

- Memilih jenis akses jaringan Anda
- Mengakses S3 Anda di ember dan objek Outposts
- Antarmuka jaringan elastis lintas akun

Memilih jenis akses jaringan Anda

Anda dapat mengakses S3 di Outposts dari dalam VPC atau dari jaringan lokal Anda. Anda berkomunikasi dengan bucket Outpost Anda dengan menggunakan titik akses dan koneksi titik akhir. Koneksi ini menjaga lalu lintas antara VPC Anda dan S3 Anda di bucket Outposts dalam jaringan. AWS Saat Anda membuat titik akhir, Anda harus menentukan jenis akses titik akhir sebagai (untuk perutean VPC) atau `Private` `CustomerOwnedIp` (untuk kumpulan alamat IP milik pelanggan [CoIP pool]).

- `Private`(untuk perutean VPC) - Jika Anda tidak menentukan jenis akses, S3 di Outposts menggunakan secara default. `Private` Dengan tipe `Private` akses, instance di VPC Anda tidak memerlukan alamat IP publik untuk berkomunikasi dengan sumber daya di Outpost Anda. Anda dapat bekerja dengan S3 di Outposts dari dalam VPC. Jenis titik akhir ini dapat diakses dari jaringan lokal Anda melalui perutean VPC langsung. Untuk informasi selengkapnya, lihat Tabel rute gateway lokal di Panduan Pengguna AWS Outposts.
- `CustomerOwnedIp`(untuk kumpulan CoIP) - Jika Anda tidak default ke jenis `Private` akses dan memilih`CustomerOwnedIp`, Anda harus menentukan rentang alamat IP. Anda dapat menggunakan jenis akses ini untuk bekerja dengan S3 di Outposts baik dari jaringan lokal maupun dalam VPC. Saat mengakses S3 di Outposts dalam VPC, lalu lintas Anda terbatas pada bandwidth gateway lokal.

Mengakses S3 Anda di ember dan objek Outposts

Untuk mengakses S3 Anda di bucket Outposts dan objek, Anda harus memiliki berikut ini:

- Titik akses untuk VPC.
- Titik akhir untuk VPC yang sama.
- Koneksi aktif antara Outpost Anda dan AndaWilayah AWS. Untuk informasi selengkapnya tentang cara menghubungkan Pos Luar Anda ke Wilayah, lihat Konektivitas pos terdepan ke AWS Wilayah di Panduan Pengguna AWSOutposts.

Untuk informasi selengkapnya tentang mengakses bucket dan objek di S3 di Outposts, lihat dan. Bekerja dengan S3 di bucket Outposts Bekerja dengan S3 di objek Outposts

Antarmuka jaringan elastis lintas akun

Titik akhir S3 di Outposts diberi nama resource dengan Amazon Resource Names (ARN). Saat titik akhir ini dibuat, siapkan beberapa AWS Outposts antarmuka jaringan elastis lintas akun. Antarmuka jaringan elastis lintas akun S3 di Outposts seperti antarmuka jaringan lainnya dengan satu pengecualian: S3 di Outposts mengaitkan antarmuka jaringan elastis lintas akun ke instans Amazon EC2.

Beban S3 on Outposts Domain Name System (DNS) menyeimbangkan permintaan Anda melalui antarmuka elastis network lintas akun. S3 on Outposts membuat cross-account elastic network interface di akun AWS Anda yang terlihat dari panel Network interface dari konsol Amazon EC2.

Untuk endpoint yang menggunakan tipe akses pool CoIP, S3 on Outposts mengalokasikan dan mengaitkan alamat IP dengan cross-account elastic network interface dari pool CoIP yang dikonfigurasi.

Bekerja dengan S3 di bucket Outposts

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di AndaAWS Outposts dan dengan mudah menyimpan dan mengambil objek di premise untuk aplikasi yang memerlukan akses data lokal, pemrosesan data lokal, dan residensi data. S3 di Outposts menyediakan kelas penyimpanan baru, S3 Outposts (`OUTPOSTS`), yang menggunakan API Amazon S3, dan dirancang untuk menyimpan data secara tahan lama dan redundan di beberapa perangkat dan server di AndaAWS Outposts. Anda dapat menggunakan API dan fitur yang sama di bucket Outpost seperti yang Anda lakukan di Amazon S3, seperti kebijakan akses, enkripsi, dan pemberian tag. Untuk informasi selengkapnya, lihat Apa itu Amazon S3 di Outposts?

Anda berkomunikasi dengan bucket Outpost Anda dengan menggunakan titik akses dan koneksi titik akhir melalui cloud pribadi virtual (VPC). Untuk mengakses S3 Anda di bucket dan objek, Outposts titik akses untuk VPC dan titik akhir untuk bucket dan objek, Anda harus memiliki titik akses untuk VPC dan objek, Anda harus memiliki titik akses untuk VPC dan titik akhir untuk VPC yang sama. Untuk informasi selengkapnya, lihat Jaringan untuk S3 di Outposts.

Bucket

Dalam S3 di Outposts, nama bucket unik untuk Outpost dan mewajibkanWilayah AWS kode untuk Wilayah yang menjadi homed,Akun AWS ID, id, Outpost id, dan nama bucket untuk mengidentifikasinya.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/bucket/bucket-name
```

Untuk informasi selengkapnya, lihat ARN sumber daya untuk S3 di Outposts.

Titik Akses

Amazon S3 di Outposts mendukung titik akses khusus virtual private cloud (VPC) sebagai satu-satunya cara untuk mengakses bucket Outposts Anda.

Titik akses menyederhanakan pengelolaan akses data dalam skala besar untuk set data bersama di Amazon S3. Titik akses diberi nama titik akhir jaringan yang dilampirkan ke bucket yang dapat Anda gunakan untuk melakukan operasi objek Amazon S3, seperti `GetObject` dan `PutObject`. Dengan S3 di bucket Outposts, Anda harus menggunakan titik akses untuk mengakses objek apapun dalam bucket Outposts. Titik akses hanya mendukung virtual-host-style pengalamatan.

Contoh berikut menunjukkan format ARN yang Anda gunakan untuk S3 di titik akses Outposts. Jalur akses ARN termasukWilayah AWS kode untuk Wilayah Outpost adalah homed ke,Akun AWS ID, Outpost ID, dan nama access point.

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Titik akhir

Untuk perutean permintaan ke S3 di titik akses Outposts, Anda harus membuat dan mengonfigurasi S3 di titik akhir Outposts. Dengan S3 on Outposts endpoint, Anda dapat menghubungkan VPC Anda secara pribadi ke bucket Outpost Anda. S3 pada titik akhir Outposts adalah pengidentifikasi sumber daya seragam virtual (URI) dari titik masuk ke S3 Anda pada ember Outposts. Mereka merupakan komponen VPC skala horizontal, redundan, dan sangat tersedia.

Setiap cloud pribadi virtual (VPC) di Outpost Anda dapat memiliki satu titik akhir terkait, dan Anda dapat memiliki hingga 100 titik akhir per Outpost. Anda harus membuat titik akhir ini agar dapat mengakses bucket Outpost Anda dan melakukan operasi objek. Membuat titik akhir ini juga memungkinkan model dan perilaku API sama dengan memungkinkan operasi yang sama untuk bekerja di S3 dan S3 di Outposts.

operasi API di S3 di S3 di Outposts

Untuk mengelola operasi API bucket Outposts, S3 di titik akhir terpisah yang berbeda dari titik akhir Amazon S3. Titik akhir ini adalah `s3-outposts.region.amazonaws.com`.

Untuk menggunakan operasi API Amazon S3, Anda harus menandatangani bucket dan objek menggunakan format ARN yang benar. Anda harus meneruskan ARN ke operasi API sehingga Amazon S3 dapat menentukan apakah permintaan tersebut untuk Amazon S3 (`s3-control.`*`region`*`.amazonaws.com`) atau untuk S3 on Outposts (`s3-outposts.`*`region`*`.amazonaws.com`). Berdasarkan format ARN, S3 kemudian dapat menandatangani dan merutekan permintaan dengan tepat.

Setiap kali permintaan dikirim ke bidang kontrol Amazon S3, SDK mengekstraksi komponen dari ARN dan mencakup header tambahan`x-amz-`*`outpost-id`*, dengan*`outpost-id`* nilai yang diambil dari ARN. Nama layanan dari ARN digunakan untuk menandai permintaan sebelum diarahkan ke S3 di titik akhir Outposts. Perilaku ini berlaku untuk semua operasi API yang ditangani oleh`s3control` klien.

Tabel berikut mencantumkan operasi API yang diperluas untuk Amazon S3 di Outposts dan perubahannya yang relatif terhadap Amazon S3.

API	S3 pada nilai parameter Outposts
`CreateBucket`	nama sebagai ARN, id Outpost
`ListRegionalBuckets`	ID Outpost
`DeleteBucket`	nama sebagai ARN
`DeleteBucketLifecycleConfiguration`	nama sebagai ARN
`GetBucketLifecycleConfiguration`	nama sebagai ARN
`PutBucketLifecycleConfiguration`	nama sebagai ARN
`GetBucketPolicy`	nama sebagai ARN
`PutBucketPolicy`	nama sebagai ARN
`DeleteBucketPolicy`	nama sebagai ARN
`GetBucketTagging`	nama sebagai ARN

API	S3 pada nilai parameter Outposts
`PutBucketTagging`	nama sebagai ARN
`DeleteBucketTagging`	nama sebagai ARN
`CreateAccessPoint`	Nama titik akses sebagai ARN
`DeleteAccessPoint`	Nama titik akses sebagai ARN
`GetAccessPoint`	Nama titik akses sebagai ARN
`GetAccessPoint`	Nama titik akses sebagai ARN
`ListAccessPoints`	Nama titik akses sebagai ARN
`PutAccessPointPolicy`	Nama titik akses sebagai ARN
`GetAccessPointPolicy`	Nama titik akses sebagai ARN
`DeleteAccessPointPolicy`	Nama titik akses sebagai ARN

Membuat dan mengelola S3 di bucket Outposts

Untuk informasi selengkapnya tentang membuat dan mengelola S3 di bucket Outposts, lihat topik berikut.

Topik

- Membuat bucket S3 di Outposts
- Menambahkan tag untuk bucket S3 di Outposts
- Mengelola akses ke bucket Amazon S3 di Outposts menggunakan kebijakan bucket
- Buat Daftar Amazon S3 di bucket Outposts
- Mendapatkan bucket S3 Outposts menggunakanAWS CLI dan SDK for Java
- Menghapus Amazon S3 di bucket Outposts
- Bekerja dengan titik akses Amazon S3 di Outposts
- Bekerja dengan Amazon S3 di titik akhir Outposts

Membuat bucket S3 di Outposts

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di Outposts AWS Anda dan dengan mudah menyimpan dan mengambil objek on-premise untuk aplikasi yang memerlukan akses data lokal, pengolahan data lokal, dan residensi data. S3 di Outposts menyediakan kelas penyimpanan baru, S3 Outposts (`OUTPOSTS`), yang menggunakan API Amazon S3, dan dirancang untuk menyimpan data secara tahan lama dan berlebihan di beberapa perangkat dan server diAWS Outposts. Anda berkomunikasi dengan bucket Outpost Anda dengan menggunakan titik akses dan koneksi endpoint melalui cloud pribadi virtual (VPC). Anda dapat menggunakan API dan fitur yang sama pada bucket Outpost seperti yang Anda lakukan pada bucket Amazon S3, termasuk kebijakan akses, enkripsi, dan penandaan. Anda dapat menggunakan S3 di Pos Terdepan melaluiAWS Management Console,AWS Command Line Interface(AWS CLI),AWSSDK, atau REST API. Untuk informasi selengkapnya, lihat Apa itu Amazon S3 di Outposts?

> **Note**
>
> Akun AWS yang menciptakan bucket memilikinya dan merupakan satu-satunya yang dapat melakukan tindakan untuk itu. Bucket memiliki properti konfigurasi, seperti Outpost, tag, enkripsi default, dan pengaturan titik akses. Pengaturan titik akses termasuk cloud pribadi virtual (VPC), kebijakan titik akses untuk mengakses objek di bucket, dan metadata lainnya. Untuk informasi selengkapnya, lihat S3 di spesifikasi Outposts.
> Jika Anda ingin membuat bucket yang menggunakanAWS PrivateLinkuntuk menyediakan akses manajemen bucket dan endpoint melaluiantarmuka VPC endpointdi virtual private cloud (VPC) Anda, lihatAWS PrivateLinkuntuk S3 di Pos Terdepan.

Contoh berikut menunjukkan cara membuat bucket S3 di Outposts dengan menggunakanAWS Management Console,AWS Command Line Interface(AWS CLI), danAWS SDK for Java.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilihEmber pos terdepan.
3. Pilih Buat bucket Outposts.
4. UntukNama ember, masukkan nama yang sesuai dengan Domain Name System (DNS) untuk bucket Anda.

Nama kelompok harus:

- Jadilah unik dalamAkun AWS, Pos Terdepan, danWilayah AWSbahwa pos terdepan adalah homed ke.
- Panjangnya 3—63 karakter.
- Tidak mengandung karakter huruf besar.
- Mulai dengan huruf atau angka kecil.

Setelah membuat bucket, Anda tidak dapat mengubah namanya. Untuk informasi tentang penamaan bucket, lihat Peraturan penamaan bucket.

> **Important**
>
> Hindari menyertakan informasi sensitif seperti nomor rekening dalam nama bucket. Nama bucket terlihat di URL yang mengarah ke objek di bucket.

5. UntukPos terdepan, pilih Outpost di mana Anda ingin ember berada.
6. Di bawahPembuatan Versi Bucket, atur status Versi S3 untuk bucket S3 on Outposts Anda ke salah satu opsi berikut:

   - Nonaktifkan(default) - Bucket tetap tidak berversi.
   - Aktifkan- Memungkinkan S3 Versioning untuk objek dalam bucket. Semua objek yang ditambahkan ke bucket menerima ID versi unik.

   Untuk informasi selengkapnya tentang S3 Versioning, lihat Mengelola versioning.
7. (Opsional) Tambahkantag opsionalbahwa Anda ingin mengasosiasikan dengan ember Outposts. Anda dapat menggunakan tag untuk melacak kriteria untuk masing-masing proyek atau grup proyek, atau untuk memberi label bucket Anda dengan menggunakan tag alokasi biaya.

   Secara default, semua objek yang disimpan di bucket Outposts Anda disimpan dengan menggunakan enkripsi sisi server dengan kunci enkripsi terkelola Amazon S3 (SSE-S3). Anda juga dapat secara eksplisit memilih untuk menyimpan objek dengan menggunakan enkripsi sisi server dengan kunci enkripsi yang disediakan pelanggan (SSE-C). Untuk mengubah jenis enkripsi, Anda harus menggunakan REST API,AWS Command Line Interface(AWS CLI), atauAWSSDK.
8. Di bagian Pengaturan titik akses Outposts, masukkan nama titik akses.

Titik akses S3 di Outposts menyederhanakan pengelolaan akses data dalam skala besar untuk kumpulan data bersama di S3 di Outposts. Titik akses diberi nama titik akhir jaringan yang dilampirkan ke bucket Outposts yang dapat Anda gunakan untuk melakukan operasi objek S3. Untuk informasi selengkapnya, lihat Titik Akses.

Nama titik akses harus unik dalam akun untuk Wilayah dan Outposts ini, dan patuh dengan Pembatasan dan batasan titik akses.

9. Pilih VPC untuk titik akses Amazon S3 di Outposts ini.

   Jika Anda tidak memiliki VPC, pilihBuat VPC. Untuk informasi selengkapnya, lihat Membuat titik akses terbatas pada cloud privat virtual.

   Virtual private cloud (VPC) memungkinkan Anda meluncurkan sumber daya AWS ke jaringan virtual yang Anda tentukan. Jaringan virtual ini sangat mirip dengan jaringan tradisional yang akan Anda operasikan di pusat data Anda sendiri, dengan manfaat dari penggunaan infrastruktur AWS yang dapat diskalakan.

10. (Opsional untuk VPC yang ada) PilihSubnet titik akhiruntuk endpoint Anda.

    Subnet adalah serangkaian alamat IP di VPC Anda. Jika Anda tidak memiliki subnet yang Anda inginkan, pilih Buat subnet. Untuk informasi selengkapnya, lihat Jaringan untuk S3 di Outposts.

11. (Opsional untuk VPC yang ada) PilihGrup keamanan titik akhiruntuk endpoint Anda.

    SEBUAHkelompok keamananbertindak sebagai firewall virtual untuk mengontrol lalu lintas masuk dan keluar.

12. (Opsional untuk VPC yang ada) PilihJenis akses titik akhir:

    - Pribadi- Untuk digunakan dengan VPC.
    - IP milik pelanggan— Untuk digunakan dengan kumpulan alamat IP milik pelanggan (CoIP pool) dari dalam jaringan lokal Anda.

13. (Opsional) TentukanKebijakan jalur akses pos terdepan. Konsol secara otomatis menampilkan Amazon Resource Name (ARN) untuk titik akses, yang dapat Anda gunakan dalam kebijakan.

14. Pilih Buat bucket Outposts.

> **Note**
>
> Diperlukan waktu hingga 5 menit agar titik akhir Outpost Anda dibuat dan bucket Anda siap digunakan. Untuk mengkonfigurasi pengaturan bucket tambahan, pilih Melihat detail.

Menggunakan AWS CLI

Example

Contoh berikut membuat S3 pada Outposts ember (`s3-outposts:CreateBucket`) dengan menggunakanAWS CLI. Untuk menjalankan perintah ini, ganti*user input placeholders*dengan informasi Anda sendiri.

```
aws s3control create-bucket --bucket example-outposts-bucket --outpost-
id op-01ac5d28a6a232904
```

MenggunakanAWSSDK untuk Java

Example

Contoh berikut membuat S3 pada Outposts ember (`s3-outposts:CreateBucket`) dengan menggunakan SDK untuk Java.

```
import com.amazonaws.services.s3control.model.*;

public String createBucket(String bucketName) {

    CreateBucketRequest reqCreateBucket = new CreateBucketRequest()
            .withBucket(bucketName)
            .withOutpostId(OutpostId)
            .withCreateBucketConfiguration(new CreateBucketConfiguration());

    CreateBucketResult respCreateBucket =
 s3ControlClient.createBucket(reqCreateBucket);
    System.out.printf("CreateBucket Response: %s%n", respCreateBucket.toString());

    return respCreateBucket.getBucketArn();
```

```
}
```

Menambahkan tag untuk bucket S3 di Outposts

Anda dapat menambahkan tag untuk bucket Amazon S3 pada Outposts Anda untuk melacak biaya penyimpanan dan kriteria lain untuk proyek individu atau kelompok proyek.

> **Note**
>
> Akun AWS yang menciptakan bucket memilikinya dan merupakan satu-satunya yang dapat mengganti tag nya.

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih bucket Outposts.
3. Pilih bucket Outposts yang tag ingin Anda edit.
4. Pilih tab Properti.
5. Di bawah Tag, pilih Edit.
6. Pilih Tambahkan tag baru, dan masukkan Kunci dan Nilai opsional.

   Tambahkan tag apa pun yang ingin Anda kaitkan dengan bucket Outposts untuk melacak kriteria lain untuk proyek individu atau grup proyek.
7. Pilih Save changes (Simpan perubahan).

Menggunakan perintah AWS CLI

AWS CLIContoh berikut menerapkan konfigurasi penandaan ke bucket S3 on Outposts dengan menggunakan dokumen JSON di folder saat ini yang menentukan tag (`tagging.json`). Untuk menggunakan contoh ini, ganti masing-masing`user input placeholder` dengan informasi Anda sendiri.

```
aws s3control put-bucket-tagging --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --tagging file://tagging.json
```

```
tagging.json

{
   "TagSet": [
     {
       "Key": "organization",
       "Value": "marketing"
     }
   ]
}
```

AWS CLIContoh berikut menerapkan konfigurasi penandaan ke bucket S3 on Outposts langsung dari baris perintah.

```
aws s3control put-bucket-tagging --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --tagging 'TagSet=[{Key=organization,Value=marketing}]'
```

Untuk informasi lebih lanjut tentang perintah ini, lihat put-bucket-taggingdi AWS CLIReferensi.

Mengelola akses ke bucket Amazon S3 di Outposts menggunakan kebijakan bucket

Kebijakan bucket adalah kebijakan AWS Identity and Access Management (IAM) berbasis sumber daya yang dapat Anda gunakan untuk memberikan izin akses ke bucket dan objek di dalamnya. Hanya pemilik bucket yang dapat mengaitkan kebijakan dengan bucket. Izin yang dipasang pada bucket berlaku untuk semua objek di bucket yang dimiliki oleh pemilik bucket. Kebijakan bucket dibatasi hingga ukuran 20 KB. Untuk informasi selengkapnya, lihat Kebijakan bucket.

Anda dapat memperbarui kebijakan bucket untuk mengelola akses ke bucket Amazon S3 di Outposts. Untuk informasi selengkapnya, lihat topik berikut.

Topik

- Menambahkan atau mengedit kebijakan bucket untuk bucket Amazon S3 di Outposts
- Melihat kebijakan bucket Amazon S3 di Outposts
- Menghapus kebijakan bucket untuk bucket Amazon S3 di Outposts
- Contoh kebijakan bucket

Menambahkan atau mengedit kebijakan bucket untuk bucket Amazon S3 di Outposts

Kebijakan bucket adalah kebijakan berbasis sumber dayaAWS Identity and Access Management (IAM) yang dapat Anda gunakan untuk memberikan izin akses ke bucket Anda dan objek di dalamnya. Hanya pemilik bucket yang dapat mengaitkan kebijakan dengan bucket. Izin yang dilampirkan pada bucket berlaku untuk semua objek di bucket yang dimiliki oleh pemilik bucket. Kebijakan ember dibatasi hingga ukuran 20 KB. Untuk informasi selengkapnya, lihat Kebijakan bucket.

Topik berikut menunjukkan cara memperbarui kebijakan bucket Amazon S3 on Outposts dengan menggunakanAWS Management Console,AWS Command Line Interface (AWS CLI), atauAWS SDK for Java.

Menggunakan konsol S3

Untuk membuat atau mengedit kebijakan bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih bucket Outposts.
3. Pilih Outposts yang kebijakan bucket yang ingin Anda edit.
4. Pilih tab Izin.
5. Di bagian kebijakan bucket Outposts, untuk membuat atau mengedit kebijakan baru, pilih Edit.

   Sekarang Anda dapat menambah atau mengedit kebijakan bucket S3 di Outposts. Untuk informasi selengkapnya, lihat Mengatur IAM dengan S3 di Outposts.

Menggunakan AWS CLI

AWS CLIContoh berikut menempatkan kebijakan pada ember Outposts.

1. Simpan kebijakan bucket berikut ke berkas JSON. Dalam contoh ini, berkas diberi namapolicy1.json. Ganti*user input placeholders* dengan informasi Anda sendiri.

```
{
   "Version":"2012-10-17",
   "Id":"testBucketPolicy",
   "Statement":[
      {
```

```
         "Sid":"st1",
         "Effect":"Allow",
         "Principal":{
            "AWS":"123456789012"
         },
         "Action":"s3-outposts:*",
         "Resource":"arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket"
      }
   ]
}
```

2. Kirim berkas JSON sebagai bagian dari perintah`put-bucket-policy` CLI. Untuk menjalankan perintah ini, ganti*`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control put-bucket-policy --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --policy file://policy1.json
```

MenggunakanAWS SDK for Java

Contoh SDK for Java berikut menempatkan kebijakan pada bucket Outposts.

```
import com.amazonaws.services.s3control.model.*;

public void putBucketPolicy(String bucketArn) {

    String policy = "{\"Version\":\"2012-10-17\",\"Id\":\"testBucketPolicy\",
\"Statement\":[{\"Sid\":\"st1\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"" +
 AccountId+ "\"},\"Action\":\"s3-outposts:*\",\"Resource\":\"" + bucketArn + "\"}]}";

    PutBucketPolicyRequest reqPutBucketPolicy = new PutBucketPolicyRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn)
            .withPolicy(policy);

    PutBucketPolicyResult respPutBucketPolicy =
 s3ControlClient.putBucketPolicy(reqPutBucketPolicy);
    System.out.printf("PutBucketPolicy Response: %s%n",
 respPutBucketPolicy.toString());
```

```
}
```

Melihat kebijakan bucket Amazon S3 di Outposts

Kebijakan bucket adalah kebijakan berbasis sumber dayaAWS Identity and Access Management (IAM) yang dapat Anda gunakan untuk memberikan izin akses ke bucket Anda dan objek di dalamnya. Hanya pemilik bucket yang dapat mengaitkan kebijakan dengan bucket. Izin yang dilampirkan pada bucket berlaku untuk semua objek di bucket yang dimiliki oleh pemilik bucket. Kebijakan ember dibatasi hingga ukuran 20 KB. Untuk informasi selengkapnya, lihat Kebijakan bucket.

Topik berikut menunjukkan cara melihat kebijakan bucket Amazon S3 di Outposts dengan menggunakanAWS Management Console,AWS Command Line Interface (AWS CLI), atauAWS SDK for Java.

Menggunakan konsol S3

Untuk membuat atau mengedit kebijakan bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi di sebelah kiri, pilih Outposts bucket.
3. Pilih bucket Outposts yang izinnya ingin Anda edit.
4. Pilih tab Izin.
5. Di bagian kebijakan bucket Outposts, Anda dapat meninjau kebijakan bucket yang ada. Untuk informasi selengkapnya, lihat Mengatur IAM dengan S3 di Outposts.

Menggunakan AWS CLI

Contoh AWS CLI berikut mendapatkan kebijakan untuk bucket Outposts. Untuk menjalankan perintah ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
aws s3control get-bucket-policy --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket
```

MenggunakanAWS SDK for Java

Contoh SDK untuk Java berikut mendapatkan kebijakan untuk bucket Outposts.

```
import com.amazonaws.services.s3control.model.*;

public void getBucketPolicy(String bucketArn) {

    GetBucketPolicyRequest reqGetBucketPolicy = new GetBucketPolicyRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn);

    GetBucketPolicyResult respGetBucketPolicy =
 s3ControlClient.getBucketPolicy(reqGetBucketPolicy);
    System.out.printf("GetBucketPolicy Response: %s%n",
 respGetBucketPolicy.toString());

}
```

Menghapus kebijakan bucket untuk bucket Amazon S3 di Outposts

Kebijakan bucket adalah kebijakan berbasis sumber dayaAWS Identity and Access Management (IAM) yang dapat Anda gunakan untuk memberikan izin akses ke bucket Anda dan objek di dalamnya. Hanya pemilik bucket yang dapat mengaitkan kebijakan bucket. Izin yang dilampirkan pada bucket berlaku untuk semua objek dalam bucket yang dimiliki oleh pemilik bucket. Kebijakan ember dibatasi hingga ukuran 20 KB. Untuk informasi selengkapnya, lihat Kebijakan bucket.

Topik berikut menunjukkan cara melihat kebijakan bucket Amazon S3 di Outposts dengan menggunakanAWS Management Console orAWS Command Line Interface (AWS CLI).

Menggunakan konsol S3

Untuk menghapus kebijakan bucket

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilih bucket Outposts.
3. Pilih bucket Outposts yang izinnya ingin Anda edit.
4. Pilih tab Izin.
5. Di bagian Kebijakan bucket Outposts, pilih Hapus.
6. Konfirmasi penghapusan.

Menggunakan AWS CLI

Contoh berikut menghapus kebijakan bucket untuk S3 on Outposts bucket (`s3-outposts:DeleteBucket`) dengan menggunakanAWS CLI. Untuk menjalankan perintah ini, ganti*`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control delete-bucket-policy --account-id 123456789012 --bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket
```

Contoh kebijakan bucket

Dengan kebijakan bucket S3 on Outposts, Anda dapat mengamankan akses ke objek di bucket S3 di Outposts, sehingga hanya pengguna dengan izin yang sesuai yang dapat mengaksesnya. Anda bahkan dapat mencegah pengguna yang diautentikasi tanpa izin yang sesuai untuk mengakses sumber daya S3 Anda di Outposts.

Bagian ini menyajikan contoh kasus penggunaan umum untuk kebijakan bucket S3 pada Outposts. Untuk menguji kebijakan ini, ganti *`user input placeholders`* dengan informasi Anda sendiri (seperti nama bucket Anda).

Untuk memberikan atau menolak izin ke satu set objek, Anda dapat menggunakan karakter wildcard (`*`) di Amazon Resource Names (ARN) dan nilai-nilai lainnya. Misalnya, Anda dapat mengontrol akses ke kelompok objek yang dimulai dengan prefiks umum atau diakhiri dengan ekstensi yang diberikan, seperti`.html`.

Untuk informasi selengkapnya tentang bahasa kebijakan AWS Identity and Access Management (IAM), lihatMengatur IAM dengan S3 di Outposts.

> **Note**
>
> Saat menguji s3outpostsizin menggunakan konsol Amazon S3, Anda harus memberikan izin tambahan yang diperlukan konsol, `s3outposts:createendpoint` seperti,, dan `s3outposts:listendpoints` sebagainya.

Sumber daya tambahan untuk membuat kebijakan bucket

- Untuk daftar tindakan kebijakan IAM, sumber daya, dan kunci kondisi yang dapat Anda gunakan saat membuat kebijakan bucket S3 di Outposts, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon S3 di Outposts.
- Untuk panduan cara membuat kebijakan S3 tentang Outposts Anda, lihat. Menambahkan atau mengedit kebijakan bucket untuk bucket Amazon S3 di Outposts

Topik

- Mengelola akses ke bucket Amazon S3 di Outposts berdasarkan alamat IP tertentu

Mengelola akses ke bucket Amazon S3 di Outposts berdasarkan alamat IP tertentu

Kebijakan bucket adalah kebijakan AWS Identity and Access Management (IAM) berbasis sumber daya yang dapat Anda gunakan untuk memberikan izin akses ke bucket dan objek di dalamnya. Hanya pemilik bucket yang dapat mengaitkan kebijakan dengan bucket. Izin yang dipasang pada bucket berlaku untuk semua objek di bucket yang dimiliki oleh pemilik bucket. Kebijakan bucket dibatasi hingga ukuran 20 KB. Untuk informasi selengkapnya, lihat Kebijakan bucket.

Membatasi akses ke suatu Wilayah tertentu

Contoh berikut menyangkal semua pengguna melakukan operasi S3 pada Outposts pada objek dalam bucket yang ditentukan kecuali permintaan tersebut berasal dari rentang alamat IP yang ditentukan.

> **Note**
>
> Saat membatasi akses ke alamat IP tertentu, pastikan Anda juga menentukan titik akhir VPC, alamat IP sumber VPC, atau alamat IP eksternal yang dapat mengakses bucket S3 on Outposts. Jika tidak, Anda mungkin kehilangan akses ke bucket jika kebijakan Anda menolak semua pengguna melakukan s3outpostsoperasi apa pun pada objek di bucket S3 di Outposts tanpa izin yang tepat.

`Condition`Pernyataan kebijakan ini mengidentifikasi *`192.0.2.0/24`*sebagai rentang alamat IP IP versi 4 (IPv4) yang diizinkan.

Pemblokiran `Condition` menggunakan kunci kondisi `NotIpAddress` dan kondisi `aws:SourceIp`, yang merupakan kunci kondisi di seluruh AWS. Kunci kondisi `aws:SourceIp` hanya dapat digunakan untuk rentang alamat IP publik. Untuk informasi selengkapnya tentang kunci kondisi ini, lihat Tindakan, sumber daya, dan kunci kondisi untuk S3 di Outposts. Nilai `aws:SourceIp` IPv4 menggunakan notasi CIDR standar. Untuk informasi selengkapnya, lihat referensi elemen kebijakan IAM JSON di Panduan Pengguna IAM.

> **Warning**
>
> Sebelum menggunakan kebijakan S3 on Outposts ini, ganti *192.0.2.0/24*rentang alamat IP dalam contoh ini dengan nilai yang sesuai untuk kasus penggunaan Anda. Jika tidak, Anda akan kehilangan kemampuan untuk mengakses bucket Anda.

```
{
    "Version": "2012-10-17",
    "Id": "S3OutpostsPolicyId1",
    "Statement": [
        {
            "Sid": "IPAllow",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3outposts:*",
            "Resource": [
                "arn:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
accesspoint/EXAMPLE-ACCESS-POINT-NAME"
                "arn:aws:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
bucket/DOC-EXAMPLE-BUCKET"
            ],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": "192.0.2.0/24"
                }
            }
        }
    ]
}
```

Izinkan alamat IPv4 dan IPv6

Saat Anda mulai menggunakan alamat IPv6, kami rekomendasikan agar Anda memperbarui semua kebijakan organisasi dengan rentang alamat IPv6 Anda selain dari rentang alamat IPv4 Anda yang telah ada. Melakukan hal ini akan membantu memastikan bahwa kebijakan terus berfungsi saat Anda melakukan transisi ke IPv6.

Kebijakan bucket contoh S3 on Outposts berikut menunjukkan cara menggabungkan rentang alamat IPv4 dan IPv6 untuk mencakup semua alamat IP valid organisasi Anda. Kebijakan contoh memungkinkan akses ke alamat IP contoh *192.0.2.1*dan *2001:DB8:1234:5678::1*dan menolak akses ke alamat *203.0.113.1*dan *2001:DB8:1234:5678:ABCD::1*.

Kunci kondisi `aws:SourceIp` hanya dapat digunakan untuk rentang alamat IP publik. Nilai IPv6 untuk `aws:SourceIp` harus dalam format standar CIDR. Untuk IPv6, kami support dengan menggunakan `::` untuk mewakili rentang 0s (misalnya, `2001:DB8:1234:5678::/64`). Untuk informasi selengkapnya, lihat operator kondisi alamat IP di Panduan Pengguna IAM.

> **Warning**
>
> Ganti rentang alamat IP dalam contoh ini dengan nilai yang sesuai untuk kasus penggunaan Anda sebelum menggunakan kebijakan S3 on Outposts ini. Jika tidak, Anda mungkin kehilangan kemampuan untuk mengakses bucket Anda.

```
{
    "Id": "S3OutpostsPolicyId2",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIPmix",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3outposts:*",
            "Resource": [
                "arn:aws:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
bucket/DOC-EXAMPLE-BUCKET",
                        "arn:aws:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-
ID/bucket/DOC-EXAMPLE-BUCKET/*"
            ],
            "Condition": {
                "IpAddress": {
```

```
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                },
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "203.0.113.0/24",
                        "2001:DB8:1234:5678:ABCD::/80"
                    ]
                }
            }
        }
    ]
}
```

Buat Daftar Amazon S3 di bucket Outposts

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di Outposts AWS Anda dan dengan mudah menyimpan dan mengambil objek on-premise untuk aplikasi yang memerlukan akses data lokal, pengolahan data lokal, dan residensi data. S3 di Outposts menyediakan kelas penyimpanan baru, S3 Outposts (`OUTPOSTS`), yang menggunakan API Amazon S3, dan dirancang untuk menyimpan data secara tahan lama dan berulang di beberapa perangkat dan server di AndaAWS Outposts. Anda berkomunikasi dengan bucket Outposts menggunakan titik akses dan koneksi titik akhir melalui cloud pribadi virtual (VPC). Anda dapat menggunakan API dan fitur yang sama di bucket Outposts seperti yang Anda lakukan di bucket Amazon S3, seperti kebijakan akses, enkripsi, dan pemberian tag. Anda dapat menggunakan S3 on Outposts melaluiAWS Management Console,AWS Command Line Interface (AWS CLI),AWS SDK, atau REST API. Untuk informasi selengkapnya, lihatApa itu Amazon S3 di Outposts?

Untuk informasi selengkapnya tentang bekerja dengan bucket di S3 di Outposts, lihatBekerja dengan S3 di bucket Outposts.

Contoh berikut menunjukkan cara mengembalikan daftar S3 Anda pada Bucket Outposts dengan menggunakanAWS Management Console,AWS CLI, danAWS SDK for Java.

Menggunakan konsol S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilih bucket Outposts.

3. Di bucket Outposts, tinjau daftar S3 di bucket Outposts.

Menggunakan AWS CLI

Contoh AWS CLI berikut mendapatkan daftar bucket di Outpost. Untuk menggunakan perintah ini, ganti masing-masing*user input placeholder* dengan informasi Anda sendiri. Untuk informasi lebih lanjut tentang perintah ini, lihat list-regional-bucketsdi AWS CLIReferensi.

```
aws s3control list-regional-buckets --account-id 123456789012 --outpost-
id op-01ac5d28a6a232904
```

MenggunakanAWS SDK for Java

Contoh SDK untuk Java berikut mendapatkan daftar bucket di Outpost. Untuk informasi lebih lanjut, lihat ListRegionalBuckets dalam Referensi API Layanan Penyimpanan Sederhana Amazon.

```
import com.amazonaws.services.s3control.model.*;

public void listRegionalBuckets() {

    ListRegionalBucketsRequest reqListBuckets = new ListRegionalBucketsRequest()
            .withAccountId(AccountId)
            .withOutpostId(OutpostId);

    ListRegionalBucketsResult respListBuckets =
 s3ControlClient.listRegionalBuckets(reqListBuckets);
    System.out.printf("ListRegionalBuckets Response: %s%n",
 respListBuckets.toString());

}
```

Mendapatkan bucket S3 Outposts menggunakanAWS CLI dan SDK for Java

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di Outposts AWS Anda dan dengan mudah menyimpan dan mengambil objek on-premise untuk aplikasi yang memerlukan akses data lokal, pengolahan data lokal, dan residensi data. S3 di Outposts menyediakan kelas penyimpanan baru, S3 Outposts (`OUTPOSTS`), yang menggunakan API Amazon S3, dan dirancang untuk menyimpan data secara tahan lama dan redundan di beberapa perangkat dan server di

AndaAWS Outposts. Anda berkomunikasi dengan bucket Outpost menggunakan titik akses dan koneksi titik akhir melalui cloud pribadi virtual (VPC). Anda dapat menggunakan API dan fitur yang sama di bucket Outpost seperti yang Anda lakukan di bucket Amazon S3, seperti kebijakan akses, enkripsi, dan pemberian tag. Anda dapat menggunakan S3 di Outposts melaluiAWS Management Console,AWS Command Line Interface (AWS CLI),AWS SDK, atau API REST. Untuk informasi lebih lanjut, lihatApa itu Amazon S3 di Outposts?

Contoh berikut menunjukkan cara untuk mendapatkan S3 pada Outposts ember dengan menggunakanAWS CLI danAWS SDK for Java.

> **Note**
>
> Saat Anda bekerja dengan Amazon S3 di Outposts melaluiAWS SDKAWS CLI atau, Anda memberikan titik akses ARN untuk Outpost sebagai pengganti nama bucket. Jalur akses ARN mengambil bentuk berikut, di mana*region* adalahWilayah AWS kode untuk Wilayah yang Outpost homed ke:
> `arn:aws:s3-outposts:`*`region`*`:`*`123456789012`*`:outpost/`*`op-01ac5d28a6a232904`*`/accesspoint/`*`example-outposts-access-point`*
> Untuk informasi lebih lanjut tentang S3 di Outposts ARNs, lihatARN sumber daya untuk S3 di Outposts.

Menggunakan AWS CLI

Contoh S3 di Outposts berikut mendapat bucket dengan menggunakanAWS CLI. Untuk menggunakan perintah ini, ganti masing-masing*user input placeholder* dengan informasi Anda sendiri. Untuk informasi selengkapnya tentang perintah ini, lihat get-bucket di AWS CLIReferensi.

```
aws s3control get-bucket --account-id 123456789012 --bucket "arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-bucket"
```

MenggunakanAWS SDK for Java

Contoh S3 di Outposts berikut mendapatkan bucket menggunakan SDK for Java. Untuk informasi lebih lanjut, lihat GetBucket dalam Referensi API Layanan Penyimpanan Sederhana Amazon.

```
import com.amazonaws.services.s3control.model.*;
```

```
public void getBucket(String bucketArn) {

    GetBucketRequest reqGetBucket = new GetBucketRequest()
            .withBucket(bucketArn)
            .withAccountId(AccountId);

    GetBucketResult respGetBucket = s3ControlClient.getBucket(reqGetBucket);
    System.out.printf("GetBucket Response: %s%n", respGetBucket.toString());

}
```

Menghapus Amazon S3 di bucket Outposts

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di Outposts AWS Anda dan dengan mudah menyimpan dan mengambil objek on-premise untuk aplikasi yang memerlukan akses data lokal, pengolahan data lokal, dan residensi data. S3 di Outposts menyediakan kelas penyimpanan baru, S3 Outposts (`OUTPOSTS`), yang menggunakan Amazon S3 APIs, dan dirancang untuk menyimpan data secara tahan lama dan redundan di beberapa perangkat dan server di AndaAWS Outposts. Anda berkomunikasi dengan bucket Outposts menggunakan titik akses dan koneksi titik akhir melalui cloud pribadi virtual (VPC). Anda dapat menggunakan API dan fitur yang sama di bucket Outposts seperti yang Anda lakukan di bucket Amazon S3, seperti kebijakan akses, enkripsi, dan pemberian tag. Anda dapat menggunakan S3 on Outposts melaluiAWS Management Console,AWS Command Line Interface (AWS CLI),AWS SDK, atau API REST. Untuk informasi selengkapnya, lihatApa itu Amazon S3 di Outposts?

Untuk informasi lebih lanjut tentang bekerja dengan bucket di S3 di Outposts, lihatBekerja dengan S3 di bucket Outposts.

Akun AWS yang menciptakan bucket memilikinya dan merupakan satu-satunya yang dapat menghapusnya.

> **Note**
>
> - Bucket Outposts harus kosong sebelum mereka dapat dihapus.
>
> Konsol Amazon S3 tidak mendukung tindakan objek S3 di Outposts. Untuk menghapus objek di bucket S3 di Outposts, Anda harus menggunakan API RESTAWS CLI, atauAWS SDK.

- Sebelum Anda dapat menghapus bucket Outposts, Anda harus menghapus titik akses Outposts untuk bucket. Untuk informasi selengkapnya, lihat Menghapus titik akses.
- Anda tidak dapat memulihkan bucket setelah itu dihapus.

Contoh berikut menunjukkan cara menghapus bucket S3 on Outposts dengan menggunakanAWS Management Console andAWS Command Line Interface (AWS CLI).

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih bucket Outposts.
3. Pilih bucket yang ingin Anda hapus, dan pilih Hapus.
4. Konfirmasi penghapusan.

Menggunakan AWS CLI

Contoh berikut menghapus bucket S3 on Outposts (`s3-outposts:DeleteBucket`) dengan menggunakanAWS CLI. Untuk menjalankan perintah ini, ganti*`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control delete-bucket --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket
```

Bekerja dengan titik akses Amazon S3 di Outposts

Untuk mengakses bucket Amazon S3 on Outposts Anda, Anda harus membuat dan mengonfigurasi titik akses.

Titik akses menyederhanakan pengelolaan akses data dalam skala besar untuk set data bersama di Amazon S3. Titik akses diberi nama titik akhir jaringan yang dilampirkan ke bucket yang dapat Anda gunakan untuk melakukan operasi objek Amazon S3, seperti `GetObject` dan `PutObject`. Dengan S3 on Outposts, Anda harus menggunakan titik akses untuk mengakses objek apapun dalam bucket Outposts. Titik akses hanya mendukung virtual-host-style pengalamatan.

> **Note**
>
> Akun AWS yang menciptakan bucket memilikinya dan merupakan satu-satunya yang dapat menetapkan titik akses untuk itu.

Bagian berikut menjelaskan cara membuat dan mengelola titik akses titik akses untuk bucket S3 on Outposts.

Topik

- Membuat titik akses S3 di Outposts
- Menggunakan alias untuk titik akses bucket S3 di Outposts Anda di bucket S3 di Outposts
- Melihat informasi tentang konfigurasi titik aksesnya
- Lihat daftar titik akses Amazon S3 di Outposts Anda
- Menghapus titik akses
- Menambahkan atau mengedit kebijakan titik akses
- Melihat kebijakan titik akses untuk S3 di titik akses Outposts

Membuat titik akses S3 di Outposts

Untuk mengakses bucket Amazon S3 on Outposts, Anda harus membuat dan mengonfigurasi titik akses.

Titik akses menyederhanakan pengelolaan akses data dalam skala besar untuk set data bersama di Amazon S3. Titik akses diberi nama titik akhir jaringan yang dilampirkan ke bucket yang dapat Anda gunakan untuk melakukan operasi objek Amazon S3, seperti `GetObject` dan `PutObject`. Dengan S3 on Outposts, Anda harus menggunakan titik akses untuk mengakses objek apapun dalam bucket Outposts. Titik akses hanya mendukung virtual-host-style pengalamatan.

Contoh berikut menunjukkan cara membuat S3 di Outposts access point dengan menggunakanAWS Management Console,AWS Command Line Interface (AWS CLI), danAWS SDK for Java.

> **Note**
>
> Akun AWS yang menciptakan bucket memilikinya dan merupakan satu-satunya yang dapat menetapkan titik akses untuk itu.

Menggunakan konsol S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih bucket Outposts.
3. Pilih bucket Outposts yang ingin Anda buatkan titik akses Outpostsnya.
4. Pilih tab Titik akses Outposts.
5. Di bagian Titik akses Outposts, pilih Buat titik akses Outposts.
6. Di Pengaturan titik akses Outposts, masukkan sebuah nama untuk titik akses, lalu pilih virtual private cloud (VPC) untuk titik akses.
7. Jika ingin menambahkan kebijakan untuk titik akses Anda, masukkan di bagian kebijakan titik akses Outposts.

   Untuk informasi selengkapnya, lihat Mengatur IAM dengan S3 di Outposts.

Menggunakan AWS CLI

Example

Contoh AWS CLI berikut membuat titik akses untuk bucket Outposts. Untuk menjalankan perintah ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
aws s3control create-access-point --account-id 123456789012
 --name example-outposts-access-point --bucket "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket" --vpc-configuration VpcId=example-vpc-12345
```

MenggunakanAWS SDK for Java

Example

Contoh SDK untuk Java berikut membuat titik akses untuk bucket Outposts. Untuk menggunakan contoh ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
import com.amazonaws.services.s3control.model.*;

public String createAccessPoint(String bucketArn, String accessPointName) {

    CreateAccessPointRequest reqCreateAP = new CreateAccessPointRequest()
```

```
            .withAccountId(AccountId)
            .withBucket(bucketArn)
            .withName(accessPointName)
            .withVpcConfiguration(new VpcConfiguration().withVpcId("vpc-12345"));

    CreateAccessPointResult respCreateAP =
 s3ControlClient.createAccessPoint(reqCreateAP);
    System.out.printf("CreateAccessPoint Response: %s%n", respCreateAP.toString());

    return respCreateAP.getAccessPointArn();

}
```

Menggunakan alias untuk titik akses bucket S3 di Outposts Anda di bucket S3 di Outposts

Dengan S3 di Outposts, Anda harus menggunakan titik akses untuk mengakses objek apa pun di bucket Outposts. Setiap kali Anda membuat titik akses untuk bucket, S3 on Outposts secara otomatis menghasilkan alias titik akses. Anda dapat menggunakan alias titik akses ini, bukan ARN titik akses untuk operasi bidang data apa pun. Misalnya, Anda dapat menggunakan alias titik akses untuk melakukan operasi tingkat objek seperti PUT, GET, LIST, dan lainnya. Untuk daftar operasi ini, lihatOperasi API Amazon S3 untuk mengelola objek.

Contoh berikut menunjukkan ARN dan titik akses alias untuk titik akses bernama*my-access-point*.

- Titik akses ARN —`arn:aws:s3-outposts:`*`region`*`:`*`123456789012`*`:outpost/`*`op-01ac5d28a6a232904`*`/accesspoint/`*`my-access-point`*
- Alias titik akses -*`my-access-po-o01ac5d28a6a232904e8xz5w8ijx1qzlbp3i3kuse10`*`--op-s3`

Untuk informasi lebih lanjut tentang ARN, lihat Amazon Resource Names (ARN) di Amazon Resource Names (ARN) di Referensi Umum AWS.

Untuk informasi lebih lanjut tentang alias akses, lihat topik berikut.

Topik

- Alias akses
- Menggunakan alias akses di S3 pada operasi objek Outposts

- Keterbatasan:

Alias akses

Alias access point dibuat dalam namespace yang sama dengan bucket S3 on Outposts. Saat membuat titik akses, S3 di Outposts secara otomatis menghasilkan alias akses yang tidak dapat diubah. Alias access point memenuhi semua persyaratan S3 yang valid pada nama bucket Outposts dan terdiri dari bagian-bagian berikut:

```
access point name prefix-metadata--op-s3
```

> **Note**
>
> `--op-s3`Akhiran dicadangkan untuk alias titik akses, sebaiknya Anda tidak menggunakannya untuk nama bucket atau access point. Untuk informasi lebih lanjut tentang aturan S3 di bucket Outposts, lihatBekerja dengan S3 di bucket Outposts.

Menemukan alias titik akses

Contoh berikut menunjukkan cara menemukan alias titik akses dengan menggunakan konsol Amazon S3 danAWS CLI.

Example : Temukan dan salin alias akses di Amazon S3 console

Setelah Anda membuat titik akses di konsol, Anda bisa mendapatkan alias titik akses dari kolom alias Access Point dalam daftar Access Point.

Untuk menyalin alias akses

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih titik akses Outposts.
3. Untuk menyalin alias akses, lakukan salah satu hal berikut:
   - Dalam daftar Access Points, pilih tombol opsi di sebelah nama titik akses, lalu pilih Salin titik akses alias.
   - Pilih nama titik akses. Kemudian, di bawah Ringkasan titik akses Outposts, salin alias titik akses.

Example : Buat titik akses dengan menggunakanAWS CLI dan temukan alias titik akses dalam respons

AWS CLIContoh berikut untuk`create-access-point` perintah menciptakan titik akses dan mengembalikan alias jalur akses yang dihasilkan secara otomatis. Untuk menjalankan perintah ini, ganti*`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control create-access-point --bucket example-outposts-bucket --name example-
outposts-access-point --account-id 123456789012

{
    "AccessPointArn":
    "arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/
accesspoint/example-outposts-access-point",
    "Alias": "example-outp-o01ac5d28a6a232904e8xz5w8ijx1qzlbp3i3kuse10--op-s3"
}
```

Example : Dapatkan alias titik akses dengan menggunakanAWS CLI

AWS CLIContoh berikut untuk`get-access-point` perintah mengembalikan informasi tentang titik akses yang ditentukan. Informasi ini mencakup alias titik akses. Untuk menjalankan perintah ini, ganti*`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control get-access-point --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --name example-outposts-access-point --account-id 123456789012

{
    "Name": "example-outposts-access-point",
    "Bucket": "example-outposts-bucket",
    "NetworkOrigin": "Vpc",
    "VpcConfiguration": {
        "VpcId": "vpc-01234567890abcdef"
    },
    "PublicAccessBlockConfiguration": {
        "BlockPublicAcls": true,
        "IgnorePublicAcls": true,
        "BlockPublicPolicy": true,
        "RestrictPublicBuckets": true
    },
    "CreationDate": "2022-09-18T17:49:15.584000+00:00",
    "Alias": "example-outp-o0b1d075431d83bebde8xz5w8ijx1qzlbp3i3kuse10--op-s3"
```

```
}
```

Example : Daftar titik akses untuk menemukan titik akses alias dengan menggunakanAWS CLI

AWS CLIContoh berikut untuk`list-access-points` perintah mencantumkan informasi tentang titik akses yang ditentukan. Informasi ini mencakup alias titik akses. Untuk menjalankan perintah ini, ganti*`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control list-access-points --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket

{
    "AccessPointList": [
        {
            "Name": "example-outposts-access-point",
            "NetworkOrigin": "Vpc",
            "VpcConfiguration": {
                "VpcId": "vpc-01234567890abcdef"
            },
            "Bucket": "example-outposts-bucket",
            "AccessPointArn": "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point",
            "Alias": "example-outp-o0b1d075431d83bebde8xz5w8ijx1qzlbp3i3kuse10--op-s3"
        }
    ]
}
```

Menggunakan alias akses di S3 pada operasi objek Outposts

Saat mengadopsi titik akses, Anda dapat menggunakan alias titik akses tanpa memerlukan perubahan kode yang ekstensif.

AWS CLIContoh ini menunjukkan`get-object` operasi untuk bucket S3 di Outposts. Contoh ini menggunakan titik akses alias sebagai nilai untuk`--bucket` bukan titik akses penuh ARN.

```
aws s3api get-object --bucket my-access-po-
o0b1d075431d83bebde8xz5w8ijx1qzlbp3i3kuse10--op-s3 --key  testkey  sample-object.rtf

{
```

```
    "AcceptRanges": "bytes",
    "LastModified": "2020-01-08T22:16:28+00:00",
    "ContentLength": 910,
    "ETag": "\"00751974dc146b76404bb7290f8f51bb\"",
    "VersionId": "null",
    "ContentType": "text/rtf",
    "Metadata": {}
}
```

Keterbatasan:

- Alias tidak dapat dikonfigurasi oleh pelanggan.
- Alias tidak dapat dihapus atau diubah atau dinonaktifkan pada titik akses.
- Anda tidak dapat menggunakan alias akses untuk S3 di operasi kontrol pesawat Outposts. Untuk daftar S3 di operasi kontrol pesawat Outposts, lihatOperasi Amazon S3 Control API untuk mengelola bucket.
- Alias tidak dapat digunakan dalam kebijakanAWS Identity and Access Management (IAM).

Melihat informasi tentang konfigurasi titik aksesnya

Titik akses menyederhanakan pengelolaan akses data dalam skala besar untuk set data bersama di Amazon S3. Titik akses diberi nama titik akhir jaringan yang dilampirkan ke bucket yang dapat Anda gunakan untuk melakukan operasi objek Amazon S3, seperti `GetObject` dan `PutObject`. Dengan S3 di Outposts, Anda harus menggunakan titik akses untuk mengakses objek apapun dalam bucket Outposts. Titik akses hanya virtual-host-style menangani.

Topik berikut menunjukkan kepada Anda cara mengembalikan informasi konfigurasi untuk titik akses S3 di Outposts dengan menggunakanAWS Management Console,AWS Command Line Interface(AWS CLI), danAWS SDK for Java.

Menggunakan konsol S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilihTitik akses pos-pos.
3. Pilih titik akses Outposts yang ingin Anda lihat.
4. Di bawahIkhtisar titik akses Outposts, tinjau detail konfigurasi titik akses.

Menggunakan AWS CLI

Contoh AWS CLI berikut mendapatkan titik akses untuk bucket Outposts. Ganti*user input placeholders*dengan informasi Anda sendiri.

```
aws s3control get-access-point --account-id 123456789012 --name arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-access-point
```

MenggunakanAWSSDK for Java

Contoh SDK untuk Java berikut mendapatkan titik akses untuk bucket Outposts.

```
import com.amazonaws.services.s3control.model.*;

public void getAccessPoint(String accessPointArn) {

    GetAccessPointRequest reqGetAP = new GetAccessPointRequest()
            .withAccountId(AccountId)
            .withName(accessPointArn);

    GetAccessPointResult respGetAP = s3ControlClient.getAccessPoint(reqGetAP);
    System.out.printf("GetAccessPoint Response: %s%n", respGetAP.toString());

}
```

Lihat daftar titik akses Amazon S3 di Outposts Anda

Titik akses menyederhanakan pengelolaan akses data dalam skala besar untuk set data bersama di Amazon S3. Titik akses diberi nama titik akhir jaringan yang dilampirkan ke bucket yang dapat Anda gunakan untuk melakukan operasi objek Amazon S3, seperti `GetObject` dan `PutObject`. Dengan S3 di Outposts, Anda harus menggunakan titik akses untuk mengakses objek apapun dalam bucket Outposts. Titik akses hanya mendukung virtual-host-style pengalamatan.

Topik berikut menunjukkan cara mengembalikan daftar S3 Anda pada titik akses Outposts dengan menggunakanAWS Management Console,AWS Command Line Interface (AWS CLI), danAWS SDK for Java.

Menggunakan konsol S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di panel navigasi sebelah kiri, pilih titik akses Outposts.
3. Di titik akses Outposts Anda.

Menggunakan AWS CLI

AWS CLIContoh berikut Membuat Daftar titik akses untuk bucket Outposts. Untuk menjalankan perintah ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
aws s3control list-access-points --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket
```

MenggunakanAWS SDK for Java

Contoh SDK for Java berikut membuat daftar titik akses untuk bucket Outposts.

```
import com.amazonaws.services.s3control.model.*;

public void listAccessPoints(String bucketArn) {

    ListAccessPointsRequest reqListAPs = new ListAccessPointsRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn);

    ListAccessPointsResult respListAPs = s3ControlClient.listAccessPoints(reqListAPs);
    System.out.printf("ListAccessPoints Response: %s%n", respListAPs.toString());

}
```

Menghapus titik akses

Titik akses menyederhanakan pengelolaan akses data dalam skala besar untuk set data bersama di Amazon S3. Titik akses diberi nama titik akhir jaringan yang dilampirkan ke bucket yang dapat Anda gunakan untuk melakukan operasi objek Amazon S3, seperti `GetObject` dan `PutObject`. Dengan S3 di Outposts, Anda harus menggunakan titik akses untuk mengakses objek apapun dalam bucket Outposts. Dukungan titik akses saja virtual-host-style menangani.

Contoh berikut menunjukkan cara menghapus titik akses dengan menggunakanAWS Management ConsoledanAWS Command Line Interface(AWS CLI).

Menggunakan konsol S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilihTitik akses Outposts.
3. Di bagian Titik akses Outposts, Pilih titik akses Outposts yang ingin Anda hapus.
4. Pilih Delete (Hapus).
5. Konfirmasi penghapusan.

Menggunakan AWS CLI

BerikutAWS CLIcontoh menghapus titik akses Outposts. Untuk menjalankan perintah ini, ganti*user input placeholders*dengan informasi Anda sendiri.

```
aws s3control delete-access-point --account-id 123456789012 --name arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

Menambahkan atau mengedit kebijakan titik akses

Titik akses memiliki izin dan kendali jaringan yang berbeda yang diterapkan Amazon S3 di Outposts untuk setiap permintaan yang dibuat melalui titik akses tersebut. Setiap titik akses memberlakukan kebijakan titik akses khusus yang bekerja bersama dengan kebijakan bucket yang melekat pada bucket dasar. Untuk informasi selengkapnya, lihat Titik Akses.

Topik berikut menunjukkan cara menambah atau mengedit kebijakan titik akses untuk titik akses S3 di Outposts Anda dengan menggunakanAWS Management Console,AWS Command Line Interface(AWS CLI), danAWS SDK for Java.

Menggunakan konsol S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilihEmber Outposts.
3. Pilih bucket Outposts yang ingin Anda edit titik akses Outpostsnya.
4. Pilih tab Titik akses Outposts.
5. Di bagian Titik akses Outposts, Pilih titik akses yang kebijakannya ingin Anda edit, dan pilih Edit kebijakan.

6. Menambahkan atau mengedit kebijakan di bagian Kebijakan titik akses Outposts. Untuk informasi selengkapnya, lihat Mengatur IAM dengan S3 di Outposts.

Menggunakan AWS CLI

BerikutAWS CLIcontoh menempatkan kebijakan pada titik akses Outposts.

1. Simpan kebijakan titik akses berikut ke berkas JSON. Dalam contoh ini, berkas diberi namaappolicy1.json. Ganti*user input placeholders*dengan informasi Anda sendiri.

```
{
   "Version":"2012-10-17",
   "Id":"exampleAccessPointPolicy",
   "Statement":[
      {
         "Sid":"st1",
         "Effect":"Allow",
         "Principal":{
            "AWS":"123456789012"
         },
         "Action":"s3-outposts:*",
         "Resource":"arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point
      }
   ]
}
```

2. Kirim berkas JSON sebagai bagian dariput-access-point-policyPerintah CLI. Ganti*user input placeholders*dengan informasi Anda sendiri.

```
aws s3control put-access-point-policy --account-id 123456789012 --name arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --policy file://appolicy1.json
```

MenggunakanAWSSDK for Java

Contoh SDK for Java berikut menempatkan kebijakan di titik akses Outposts.

```
import com.amazonaws.services.s3control.model.*;
```

```
public void putAccessPointPolicy(String accessPointArn) {

    String policy = "{\"Version\":\"2012-10-17\",\"Id\":\"testAccessPointPolicy\",
\"Statement\":[{\"Sid\":\"st1\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"" +
 AccountId + "\"},\"Action\":\"s3-outposts:*\",\"Resource\":\"" + accessPointArn +
 "\"}]}";

    PutAccessPointPolicyRequest reqPutAccessPointPolicy = new
 PutAccessPointPolicyRequest()
            .withAccountId(AccountId)
            .withName(accessPointArn)
            .withPolicy(policy);

    PutAccessPointPolicyResult respPutAccessPointPolicy =
 s3ControlClient.putAccessPointPolicy(reqPutAccessPointPolicy);
    System.out.printf("PutAccessPointPolicy Response: %s%n",
 respPutAccessPointPolicy.toString());
    printWriter.printf("PutAccessPointPolicy Response: %s%n",
 respPutAccessPointPolicy.toString());

}
```

Melihat kebijakan titik akses untuk S3 di titik akses Outposts

Titik akses memiliki izin dan kendali jaringan yang berbeda yang diterapkan Amazon S3 di Outposts untuk setiap permintaan yang dibuat melalui titik akses tersebut. Setiap titik akses memberlakukan kebijakan titik akses khusus yang bekerja bersama dengan kebijakan bucket yang melekat pada bucket dasar. Untuk informasi selengkapnya, lihat Titik Akses.

Untuk informasi lebih lanjut tentang titik akses di S3 di Outposts, lihatBekerja dengan S3 di bucket Outposts.

Topik berikut menunjukkan kepada Anda cara melihat kebijakan titik akses S3 di Outposts Anda dengan menggunakanAWS Management Console,AWS Command Line Interface(AWS CLI), danAWS SDK for Java.

Menggunakan konsol S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilihTitik akses pos-pos.
3. Pilih titik akses Outposts yang ingin Anda lihat.

4. PadaIzintab, meninjau kebijakan titik akses S3 di Outposts.
5. Untuk mengedit kebijakan titik akses, lihatMenambahkan atau mengedit kebijakan titik akses.

Menggunakan AWS CLI

BerikutAWS CLIContoh mendapatkan kebijakan untuk titik akses Outposts. Untuk menjalankan perintah ini, ganti*user input placeholders*dengan informasi Anda sendiri.

```
aws s3control get-access-point-policy --account-id 123456789012 --name arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

MenggunakanAWSSDK for Java

Contoh SDK for Java berikut mendapatkan kebijakan untuk titik akses Outposts.

```
import com.amazonaws.services.s3control.model.*;

public void getAccessPointPolicy(String accessPointArn) {

    GetAccessPointPolicyRequest reqGetAccessPointPolicy = new
 GetAccessPointPolicyRequest()
            .withAccountId(AccountId)
            .withName(accessPointArn);

    GetAccessPointPolicyResult respGetAccessPointPolicy =
 s3ControlClient.getAccessPointPolicy(reqGetAccessPointPolicy);
    System.out.printf("GetAccessPointPolicy Response: %s%n",
 respGetAccessPointPolicy.toString());
    printWriter.printf("GetAccessPointPolicy Response: %s%n",
 respGetAccessPointPolicy.toString());

}
```

Bekerja dengan Amazon S3 di titik akhir Outposts

Untuk perutean permintaan ke titik akses Amazon S3 di Outposts, Anda harus membuat dan mengonfigurasi S3 di titik akhir Outposts. Untuk membuat titik akhir, Anda memerlukan koneksi aktif dengan tautan layanan Anda ke wilayah asal Outposts Anda. Setiap cloud pribadi virtual (VPC) pada Outposts Anda dapat memiliki satu titik akhir terkait. Untuk informasi lebih lanjut tentang kuota akhir,

lihat S3 di persyaratan jaringan Outposts. Anda harus membuat titik akhir agar dapat mengakses bucket Outposts Anda dan melakukan operasi objek. Untuk informasi selengkapnya, lihat Titik akhir.

Setelah Anda membuat titik akhir, Anda dapat menggunakan bidang 'Status', untuk memahami keadaan titik akhir. Jika Outposts Anda offline, itu akan mengembalikan CREATE_FAILED. Anda dapat memeriksa koneksi tautan layanan, menghapus titik akhir, dan mencoba kembali operasi create setelah koneksi Anda dilanjutkan. Untuk daftar kode kesalahan tambahan, lihat di bawah. Untuk informasi selengkapnya, lihat Titik akhir.

API	Status	Kode Kesalahan Alasan Gagal	Pesan - Alasan Gagal
`CreateEndpoint`	Buat_Gagal	OutpostNotReachable	Endpoint tidak dapat dibuat karena koneksi tautan layanan ke Wilayah asal Outposts Anda sedang down. Periksa koneksi Anda, hapus titik akhir, dan coba lagi.
`CreateEndpoint`	Buat_Gagal	InternalError	Endpoint tidak dapat dibuat karena Kesalahan Internal. Harap hapus endpoint dan buat lagi.
`DeleteEndpoint`	Delete_Gagal	OutpostNotReachable	Endpoint tidak dapat dihapus karena koneksi link layanan ke Wilayah asal Outposts Anda sedang down. Periksa koneksi Anda dan coba lagi.
`DeleteEndpoint`	Delete_Gagal	InternalError	Endpoint tidak dapat dihapus karena Kesalahan Internal. Harap coba lagi.

Untuk informasi tentang bekerja dengan bucket di Outposts, lihatBekerja dengan S3 di bucket Outposts.

Bagian berikut ini menjelaskan cara membuat dan mengelola titik akhir untuk S3 di Outposts.

Topik

- Membuat titik akhir di Outpost
- Melihat daftar Amazon S3 Anda di titik akhir Outposts

- Menghapus titik akhir Amazon S3 di titik akhir Outposts

Membuat titik akhir di Outpost

Untuk merutekan permintaan ke Amazon S3 di jalur akses Outposts, Anda harus membuat dan mengonfigurasi titik akhir S3 di Outposts. Untuk membuat titik akhir, Anda akan memerlukan koneksi aktif dengan tautan layanan Anda ke wilayah rumah Outposts Anda. Setiap virtual private cloud (VPC) di Outpost Anda dapat memiliki satu titik akhir terkait. Untuk informasi lebih lanjut tentang kuota titik akhir, lihat. S3 di persyaratan jaringan Outposts Anda harus membuat endpoint untuk dapat mengakses bucket Outposts Anda dan melakukan operasi objek. Untuk informasi selengkapnya, lihat Titik akhir.

Izin

Untuk informasi selengkapnya tentang izin yang diperlukan untuk membuat titik akhir, lihat. Izin untuk titik akhir S3 di Outposts

Saat Anda membuat titik akhir, S3 di Outposts juga membuat peran terkait layanan di situs Anda. Akun AWS Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan untuk Amazon S3 di Outposts.

Contoh berikut menunjukkan cara membuat S3 pada titik akhir Outposts dengan menggunakanAWS Command Line Interface, AWS CLI (), AWS Management Console dan. AWS SDK for Java

Menggunakan konsol S3

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Jalur akses Outposts.
3. Pilih tab titik akhir Outposts.
4. Pilih Create Outposts endpoint.
5. Di bawah Outpost, pilih Outpost untuk membuat endpoint ini.
6. Di bawah VPC, pilih VPC yang belum memiliki titik akhir dan yang juga sesuai dengan aturan untuk titik akhir Outposts.

   Virtual private cloud (VPC) memungkinkan Anda meluncurkan sumber daya AWS ke jaringan virtual yang Anda tentukan. Jaringan virtual ini sangat mirip dengan jaringan tradisional yang

akan Anda operasikan di pusat data Anda sendiri, dengan manfaat dari penggunaan infrastruktur AWS yang dapat diskalakan.

Jika Anda tidak memiliki VPC, pilih Buat VPC. Untuk informasi selengkapnya, lihat Membuat titik akses terbatas pada cloud privat virtual.

7. Pilih Create Outposts endpoint.

Menggunakan AWS CLI

Example

AWS CLIContoh berikut membuat endpoint untuk Outpost dengan menggunakan jenis akses sumber daya VPC. VPC berasal dari subnet. Untuk menjalankan perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1
```

AWS CLIContoh berikut membuat endpoint untuk Outpost dengan menggunakan jenis akses pool alamat IP (CoIP pool) milik pelanggan. Untuk menjalankan perintah ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
aws s3outposts create-endpoint --outpost-id op-01ac5d28a6a232904 --subnet-id
 subnet-8c7a57c5 --security-group-id sg-ab19e0d1 --access-type CustomerOwnedIp --
customer-owned-ipv4-pool ipv4pool-coip-12345678901234567
```

Menggunakan AWS SDK for Java

Example

Contoh SDK untuk Java berikut membuat titik akhir untuk Outpost. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
import com.amazonaws.services.s3outposts.model.CreateEndpointRequest;
import com.amazonaws.services.s3outposts.model.CreateEndpointResult;

public void createEndpoint() {
```

```
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    CreateEndpointRequest createEndpointRequest = new CreateEndpointRequest()
                .withOutpostId("op-0d79779cef3c30a40")
                .withSubnetId("subnet-8c7a57c5")
                .withSecurityGroupId("sg-ab19e0d1")
                .withAccessType("CustomerOwnedIp")
                .withCustomerOwnedIpv4Pool("ipv4pool-coip-12345678901234567");
    // Use .withAccessType and .withCustomerOwnedIpv4Pool only when the access type is
    // customer-owned IP address pool (CoIP pool)
    CreateEndpointResult createEndpointResult =
 s3OutpostsClient.createEndpoint(createEndpointRequest);
    System.out.println("Endpoint is created and its ARN is " +
 createEndpointResult.getEndpointArn());
}
```

Melihat daftar Amazon S3 Anda di titik akhir Outposts

Untuk perutean permintaan ke Amazon S3 di titik akses Outposts, Anda harus membuat dan mengonfigurasi S3 di titik akhir Outposts. Untuk membuat titik akhir, Anda memerlukan koneksi aktif dengan tautan layanan Anda ke wilayah asal Outposts Anda. Setiap cloud pribadi virtual (VPC) pada Outposts Anda dapat memiliki satu titik akhir terkait. Untuk informasi lebih lanjut tentang kuota titik akhir, lihat S3 di persyaratan jaringan Outposts. Anda harus membuat titik akhir agar dapat mengakses bucket Outposts Anda dan melakukan operasi objek. Untuk informasi selengkapnya, lihat Titik akhir.

Contoh berikut menunjukkan cara mengembalikan daftar S3 Anda pada titik akhir Outposts dengan menggunakanAWS Management Console,AWS Command Line Interface (AWS CLI), danAWS SDK for Java.

Menggunakan konsol S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih titik akses Outposts.
3. Pada halaman titik akses Outposts, pilih tab Titik akhir Outposts.
4. Di bawah titik akhir Outposts, Anda dapat melihat daftar S3 Anda di endpoint Outposts.

Menggunakan AWS CLI

AWS CLIContoh berikut mencantumkan titik akhir untukAWS Outposts sumber daya yang terkait dengan akun Anda. Untuk informasi selengkapnya tentang perintah ini, lihat daftar-endpoint di AWS CLIReferensi.

```
aws s3outposts list-endpoints
```

MenggunakanAWS SDK for Java

Contoh SDK for Java berikut membuat daftar titik akhir untuk Outpost. Untuk informasi lebih lanjut, lihat ListEndpoints dalam Referensi API Layanan Penyimpanan Sederhana Amazon.

```
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
import com.amazonaws.services.s3outposts.model.ListEndpointsRequest;
import com.amazonaws.services.s3outposts.model.ListEndpointsResult;

public void listEndpoints() {
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    ListEndpointsRequest listEndpointsRequest = new ListEndpointsRequest();
    ListEndpointsResult listEndpointsResult =
 s3OutpostsClient.listEndpoints(listEndpointsRequest);
    System.out.println("List endpoints result is " + listEndpointsResult);
}
```

Menghapus titik akhir Amazon S3 di titik akhir Outposts

Untuk perutean permintaan ke Amazon S3 di titik akses Outposts, Anda harus membuat dan mengonfigurasi S3 di titik akhir Outposts. Untuk membuat titik akhir, Anda memerlukan koneksi aktif dengan tautan layanan Anda ke wilayah asal Outposts Anda. Setiap virtual private cloud (VPC) pada Outpost Anda dapat memiliki satu titik akhir terkait. Untuk informasi lebih lanjut tentang kuota titik akhir, lihat S3 di persyaratan jaringan Outposts. Anda harus membuat titik akhir agar dapat mengakses bucket Outposts Anda dan melakukan operasi objek. Untuk informasi selengkapnya, lihat Titik akhir.

Contoh berikut menunjukkan cara menghapus S3 Anda pada titik akhir Outposts dengan menggunakanAWS Management Console,AWS Command Line Interface (AWS CLI), danAWS SDK for Java.

Menggunakan konsol S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih titik akses Outposts.
3. Pada halaman titik akses Outposts, pilih tab Titik akhir Outposts.
4. Di titik akhir Outposts, pilih titik akhir yang ingin Anda hapus, lalu pilih Hapus.

Menggunakan AWS CLI

AWS CLIContoh berikut menghapus titik akhir untuk Outpost. Untuk menjalankan perintah ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
aws s3outposts delete-endpoint --endpoint-id example-endpoint-id --outpost-
id op-01ac5d28a6a232904
```

MenggunakanAWS SDK for Java

Contoh SDK for Java berikut menghapus titik akhir untuk Outpost. Untuk menggunakan contoh ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
import com.amazonaws.arn.Arn;
import com.amazonaws.services.s3outposts.AmazonS3Outposts;
import com.amazonaws.services.s3outposts.AmazonS3OutpostsClientBuilder;
import com.amazonaws.services.s3outposts.model.DeleteEndpointRequest;

public void deleteEndpoint(String endpointArnInput) {
    String outpostId = "op-01ac5d28a6a232904";
    AmazonS3Outposts s3OutpostsClient = AmazonS3OutpostsClientBuilder
                .standard().build();

    Arn endpointArn = Arn.fromString(endpointArnInput);
    String[] resourceParts = endpointArn.getResource().getResource().split("/");
    String endpointId = resourceParts[resourceParts.length - 1];
    DeleteEndpointRequest deleteEndpointRequest = new DeleteEndpointRequest()
                .withEndpointId(endpointId)
                .withOutpostId(outpostId);
    s3OutpostsClient.deleteEndpoint(deleteEndpointRequest);
    System.out.println("Endpoint with id " + endpointId + " is deleted.");
}
```

Bekerja dengan S3 di objek Outposts

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di AWS Outposts dan dengan mudah menyimpan dan mengambil objek di tempat untuk aplikasi yang memerlukan akses data lokal, pemrosesan data lokal, dan residensi data. S3 on Outposts menyediakan kelas penyimpanan baru, S3 Outposts `OUTPOSTS` (), yang menggunakan API Amazon S3, dan dirancang untuk menyimpan data secara tahan lama dan berlebihan di beberapa perangkat dan server di perangkat Anda. AWS Outposts Anda berkomunikasi dengan bucket Outposts menggunakan titik akses dan koneksi titik akhir melalui cloud privat virtual (VPC). Anda dapat menggunakan API dan fitur yang sama di bucket Outposts seperti yang Anda lakukan di bucket Amazon S3, termasuk kebijakan akses, enkripsi, dan pemberian tag. Anda dapat menggunakan S3 di Outposts melalui AWS Management Console AWS Command Line Interface ,AWS CLI(), SDK AWS , atau REST API.

Objek adalah entitas dasar yang disimpan di Amazon S3 di Outposts. Setiap objek dimuat dalam bucket. Anda harus menggunakan titik akses untuk mengakses objek apa pun dalam bucket Outposts. Saat menentukan bucket untuk operasi objek, Anda menggunakan Amazon Resource Name (ARN) titik akses atau alias titik akses. Untuk informasi selengkapnya tentang alias titik akses, lihat Menggunakan alias untuk titik akses bucket S3 di Outposts Anda di bucket S3 di Outposts.

Contoh berikut menunjukkan format ARN untuk S3 pada jalur akses Outposts, yang mencakup Wilayah AWS kode untuk Wilayah tempat Pos Luar berada, ID, Akun AWS ID Pos Luar, dan nama titik akses:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Untuk informasi selengkapnya tentang ARN S3 di Outposts, lihat ARN sumber daya untuk S3 di Outposts.

ARN objek menggunakan format berikut, yang mencakup tempat Pos Luar, Akun AWS ID, ID Pos Luar, nama bucket, dan kunci objek: Wilayah AWS

```
arn:aws:s3-outposts:us-west-2:123456789012:outpost/ op-01ac5d28a6a232904/
bucket/example-s3-bucket1/object/myobject
```

Dengan Amazon S3 di Outposts, data objek selalu disimpan di Outpost. Saat AWS memasang rak Outpost, data Anda tetap lokal di Outpost Anda untuk memenuhi persyaratan residensi data. Objek Anda tidak akan meninggalkan Outpost dan tidak berada di Wilayah AWS. Karena di-host In-region, Anda tidak dapat menggunakan konsol untuk mengunggah atau mengelola objek di Outpost Anda. AWS Management Console Namun, Anda dapat menggunakan REST API, AWS Command Line

Interface (AWS CLI), dan AWS SDK untuk mengunggah dan mengelola objek Anda melalui titik akses Anda.

Topik

- Unggah objek ke ember S3 di Outposts
- Menyalin objek di Amazon S3 pada bucket Outposts menggunakanAWS SDK for Java
- Pengambilamakan objek Amazon S3 di bucket Outposts
- Membuat daftar objek di Amazon S3 pada bucket Outposts
- Menghapus objek di Amazon S3 di bucket Outposts
- Menggunakan HeadBucket untuk menentukan apakah S3 di Outposts bucket ada dan Anda memiliki izin akses
- Melakukan dan mengelola upload multipart dengan SDK for Java
- Menggunakan URL yang telah ditandatangani untuk S3 di Outposts
- Amazon S3 di Outposts dengan Amazon EMR lokal di Outposts
- Caching otorisasi dan otentikasi

Unggah objek ke ember S3 di Outposts

Objek adalah entitas dasar yang disimpan di Amazon S3 di Outposts. Setiap objek dimuat dalam bucket. Anda harus menggunakan titik akses untuk mengakses objek apa pun dalam bucket Outposts. Saat menentukan bucket untuk operasi objek, Anda menggunakan Amazon Resource Name (ARN) titik akses atau alias titik akses. Untuk informasi selengkapnya tentang alias titik akses, lihat Menggunakan alias untuk titik akses bucket S3 di Outposts Anda di bucket S3 di Outposts.

Contoh berikut menunjukkan format ARN untuk S3 pada jalur akses Outposts, yang mencakup Wilayah AWS kode untuk Wilayah tempat Pos Luar berada, ID, Akun AWS ID Pos Luar, dan nama titik akses:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Untuk informasi selengkapnya tentang ARN S3 di Outposts, lihat ARN sumber daya untuk S3 di Outposts.

Dengan Amazon S3 di Outposts, data objek selalu disimpan di Outpost. Saat AWS memasang rak Outpost, data Anda tetap lokal di Outpost Anda untuk memenuhi persyaratan residensi data. Objek Anda tidak akan meninggalkan Outpost dan tidak berada di Wilayah AWS. Karena di-host In-region,

Anda tidak dapat menggunakan konsol untuk mengunggah atau mengelola objek di Outpost Anda. AWS Management Console Namun, Anda dapat menggunakan REST API, AWS Command Line Interface (AWS CLI), dan AWS SDK untuk mengunggah dan mengelola objek Anda melalui titik akses Anda.

Berikut AWS CLI dan AWS SDK for Java contoh menunjukkan cara mengunggah objek ke bucket S3 di Outposts dengan menggunakan titik akses.

AWS CLI

Example

Contoh berikut menempatkan objek bernama `sample-object.xml` ke bucket S3 di Outposts (`s3-outposts:PutObject`) menggunakan AWS CLI. Untuk menggunakan perintah ini, ganti masing-masing *`user input placeholder`* dengan informasi Anda sendiri. Untuk informasi selengkapnya tentang perintah ini, lihat put-object dalam Referensi AWS CLI .

```
aws s3api put-object --bucket arn:aws:s3-
outposts:Region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --key sample-object.xml --body sample-object.xml
```

SDK for Java

Example

Contoh berikut menempatkan objek dalam bucket S3 di Outposts menggunakan SDK for Java. Untuk menggunakan contoh ini, ganti masing-masing *`user input placeholder`* dengan informasi Anda sendiri. Untuk informasi selengkapnya, lihat Mengunggah Objek.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ObjectMetadata;
import com.amazonaws.services.s3.model.PutObjectRequest;

import java.io.File;

public class PutObject {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String stringObjKeyName = "*** String object key name ***";
```

```
        String fileObjKeyName = "*** File object key name ***";
        String fileName = "*** Path to file to upload ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Upload a text string as a new object.
            s3Client.putObject(accessPointArn, stringObjKeyName, "Uploaded String
 Object");

            // Upload a file as a new object with ContentType and title specified.
            PutObjectRequest request = new PutObjectRequest(accessPointArn,
 fileObjKeyName, new File(fileName));
            ObjectMetadata metadata = new ObjectMetadata();
            metadata.setContentType("plain/text");
            metadata.addUserMetadata("title", "someTitle");
            request.setMetadata(metadata);
            s3Client.putObject(request);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Menyalin objek di Amazon S3 pada bucket Outposts menggunakanAWS SDK for Java

Objek adalah entitas dasar yang disimpan di Amazon S3 di Outposts. Setiap objek dimuat dalam bucket. Anda harus menggunakan titik akses untuk mengakses objek apapun dalam bucket Outpost. Saat Anda menentukan bucket untuk operasi objek, Anda menggunakan titik akses Amazon

Resource Name (ARN) atau alias titik akses. Untuk informasi selengkapnya tentang titik akses alias, lihatMenggunakan alias untuk titik akses bucket S3 di Outposts Anda di bucket S3 di Outposts.

Contoh berikut menunjukkan format ARN untuk S3 di Outposts access points, yang mencakupWilayah AWS kode untuk Wilayah yang Outpost homed ke,Akun AWS ID, Outpost ID, dan nama access point:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Untuk informasi lebih lanjut tentang S3 di Outposts ARN, lihatARN sumber daya untuk S3 di Outposts.

Dengan Amazon S3 di Outposts, data objek selalu disimpan di Outpost. KetikaAWS menginstal rak Outpost, data Anda tetap lokal untuk Outpost Anda untuk memenuhi persyaratan data-residensi. Objek Anda tidak pernah meninggalkan Outpost Anda dan tidak dalamWilayah AWS. Karena di-host In-region, Anda tidak dapat menggunakan konsol untuk mengunggah atau mengelola objek di Outpost Anda.AWS Management Console Namun, Anda dapat menggunakan REST API,AWS Command Line Interface (AWS CLI), danAWS SDK untuk mengunggah dan mengelola objek Anda melalui titik akses Anda.

Contoh berikut menunjukkan cara menyalin objek di bucket S3 di Outposts dengan menggunakanAWS SDK for Java.

MenggunakanAWS SDK for Java

Contoh S3 di Outposts berikut menyalin objek ke objek baru di bucket yang sama dengan SDK for Java. Untuk menggunakan contoh ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.CopyObjectRequest;

public class CopyObject {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String sourceKey = "*** Source object key ***";
        String destinationKey = "*** Destination object key ***";
```

```
        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Copy the object into a new object in the same bucket.
            CopyObjectRequest copyObjectRequest = new CopyObjectRequest(accessPointArn,
 sourceKey, accessPointArn, destinationKey);
            s3Client.copyObject(copyObjectRequest);
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Pengambilamakan objek Amazon S3 di bucket Outposts

Objek adalah entitas dasar yang disimpan di Amazon S3 di Outposts. Setiap objek dimuat dalam bucket. Anda harus menggunakan titik akses untuk mengakses objek apapun dalam bucket Outposts. Saat Anda menentukan bucket untuk operasi objek, Anda menggunakan titik akses Amazon Resource Name (ARN) atau alias titik akses. Untuk informasi selengkapnya tentang titik akses alias, lihatMenggunakan alias untuk titik akses bucket S3 di Outposts Anda di bucket S3 di Outposts.

Contoh berikut menunjukkan format ARN untuk S3 di Outposts access points, yang mencakupWilayah AWS kode untuk Wilayah yang Outpost homed ke,Akun AWS ID, Outpost ID, dan nama access point:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Untuk informasi lebih lanjut tentang S3 di Outposts ARN, lihatARN sumber daya untuk S3 di Outposts.

Dengan Amazon S3 di Outposts, data objek selalu disimpan di Outpost. KetikaAWS menginstal rak Outpost, data Anda tetap lokal untuk Outpost Anda untuk memenuhi persyaratan data-residensi. Objek Anda tidak pernah meninggalkan Outpost Anda dan tidak dalamWilayah AWS. Karena di-host In-region, Anda tidak dapat menggunakan konsol untuk mengunggah atau mengelola objek di Outpost Anda.AWS Management Console Namun, Anda dapat menggunakan REST API,AWS Command Line Interface (AWS CLI), danAWS SDK untuk mengunggah dan mengelola objek Anda melalui titik akses Anda.

Contoh berikut menunjukkan cara untuk men-download (mendapatkan) objek dengan menggunakanAWS Command Line Interface (AWS CLI) danAWS SDK for Java.

Menggunakan AWS CLI

Contoh berikut mendapat sebuah objek bernama`sample-object.xml` dari S3 pada Outposts bucket (`s3-outposts:GetObject`) dengan menggunakanAWS CLI. Untuk menggunakan perintah ini, ganti masing-masing*`user input placeholder`* dengan informasi Anda sendiri. Untuk informasi selengkapnya tentang perintah ini, lihat get-object di AWS CLIReferensi.

```
aws s3api get-object --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point --key testkey sample-object.xml
```

MenggunakanAWS SDK for Java

Contoh S3 di Outposts berikut mendapatkan objek dengan menggunakan SDK for Java. Untuk menggunakan contoh ini, ganti masing-masing*`user input placeholder`* dengan informasi Anda sendiri. Untuk informasi lebih lanjut, lihat GetObject dalam Referensi API Layanan Penyimpanan Sederhana Amazon.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.GetObjectRequest;
import com.amazonaws.services.s3.model.ResponseHeaderOverrides;
import com.amazonaws.services.s3.model.S3Object;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
```

```
public class GetObject {
    public static void main(String[] args) throws IOException {
        String accessPointArn = "*** access point ARN ***";
        String key = "*** Object key ***";

        S3Object fullObject = null, objectPortion = null, headerOverrideObject = null;
        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Get an object and print its contents.
            System.out.println("Downloading an object");
            fullObject = s3Client.getObject(new GetObjectRequest(accessPointArn, key));
            System.out.println("Content-Type: " +
 fullObject.getObjectMetadata().getContentType());
            System.out.println("Content: ");
            displayTextInputStream(fullObject.getObjectContent());

            // Get a range of bytes from an object and print the bytes.
            GetObjectRequest rangeObjectRequest = new GetObjectRequest(accessPointArn,
 key)
                    .withRange(0, 9);
            objectPortion = s3Client.getObject(rangeObjectRequest);
            System.out.println("Printing bytes retrieved.");
            displayTextInputStream(objectPortion.getObjectContent());

            // Get an entire object, overriding the specified response headers, and
 print the object's content.
            ResponseHeaderOverrides headerOverrides = new ResponseHeaderOverrides()
                    .withCacheControl("No-cache")
                    .withContentDisposition("attachment; filename=example.txt");
            GetObjectRequest getObjectRequestHeaderOverride = new
 GetObjectRequest(accessPointArn, key)
                    .withResponseHeaders(headerOverrides);
            headerOverrideObject = s3Client.getObject(getObjectRequestHeaderOverride);
            displayTextInputStream(headerOverrideObject.getObjectContent());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
```

```
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        } finally {
            // To ensure that the network connection doesn't remain open, close any
 open input streams.
            if (fullObject != null) {
                fullObject.close();
            }
            if (objectPortion != null) {
                objectPortion.close();
            }
            if (headerOverrideObject != null) {
                headerOverrideObject.close();
            }
        }
    }

    private static void displayTextInputStream(InputStream input) throws IOException {
        // Read the text input stream one line at a time and display each line.
        BufferedReader reader = new BufferedReader(new InputStreamReader(input));
        String line = null;
        while ((line = reader.readLine()) != null) {
            System.out.println(line);
        }
        System.out.println();
    }
}
```

Membuat daftar objek di Amazon S3 pada bucket Outposts

Objek adalah entitas dasar yang disimpan di Amazon S3 di Outposts. Setiap objek dimuat dalam bucket. Anda harus menggunakan titik akses untuk mengakses objek apapun dalam bucket Outposts. Saat Anda menentukan bucket untuk operasi objek, Anda menggunakan titik akses Amazon Resource Name (ARN) atau alias titik akses. Untuk informasi lebih lanjut tentang titik alias titik akses, lihat[Menggunakan alias untuk titik akses bucket S3 di Outposts Anda di bucket S3 di Outposts](#).

Contoh berikut menunjukkan format ARN untuk S3 di Outposts access points, yang mencakupWilayah AWS kode untuk Wilayah yang Outpost homed ke,Akun AWS ID, Outpost ID, dan nama access point:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Untuk informasi lebih lanjut tentang S3 di Outposts ARN, lihatARN sumber daya untuk S3 di Outposts.

> **Note**
>
> Dengan Amazon S3 di Outposts, data objek selalu disimpan di Outpost. KetikaAWS menginstal rak Outpost, data Anda tetap lokal untuk Outpost Anda untuk memenuhi persyaratan data-residensi. Objek Anda tidak pernah meninggalkan Outpost Anda dan tidak dalamWilayah AWS. Karena di-host In-region, Anda tidak dapat menggunakan konsol untuk mengunggah atau mengelola objek di Outpost Anda.AWS Management Console Namun, Anda dapat menggunakan REST API,AWS Command Line Interface (AWS CLI), danAWS SDK untuk mengunggah dan mengelola objek Anda melalui titik akses Anda.

Contoh berikut menunjukkan cara untuk daftar objek dalam S3 pada Outposts ember menggunakanAWS CLI danAWS SDK for Java.

Menggunakan AWS CLI

Contoh berikut mencantumkan objek dalam S3 on Outposts bucket (`s3-outposts:ListObjectsV2`) dengan menggunakanAWS CLI. Untuk menggunakan perintah ini, ganti masing-masing*`user input placeholder`* dengan informasi Anda sendiri. Untuk informasi lebih lanjut tentang perintah ini, lihat list-objects-v2 di AWS CLIReferensi.

```
aws s3api list-objects-v2 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

> **Note**
>
> Saat menggunakan tindakan ini dengan Amazon S3 di Outposts melaluiAWS SDK, Anda menyediakan ARN titik akses Outposts

sebagai pengganti nama bucket, dalam bentuk berikut: `arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-Outposts-Access-Point`. Untuk informasi lebih lanjut tentang S3 di Outposts ARN, lihatARN sumber daya untuk S3 di Outposts.

MenggunakanAWS SDK for Java

Contoh S3 di Outposts berikut membuat daftar objek di bucket dengan menggunakan SDK for Java. Untuk menggunakan contoh ini, ganti masing-masing*user input placeholder* dengan informasi Anda sendiri.

> **Important**
>
> Contoh ini menggunakan ListObjectsV2, yang merupakan revisi terbaru dari operasi`ListObjects` API. Kami menyarankan agar Anda menggunakan operasi API yang direvisi ini untuk pengembangan aplikasi. Untuk kompatibilitas mundur, Amazon S3 terus mendukung versi sebelumnya dari operasi API ini.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListObjectsV2Request;
import com.amazonaws.services.s3.model.ListObjectsV2Result;
import com.amazonaws.services.s3.model.S3ObjectSummary;

public class ListObjectsV2 {

    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();
```

```
            System.out.println("Listing objects");

            // maxKeys is set to 2 to demonstrate the use of
            // ListObjectsV2Result.getNextContinuationToken()
            ListObjectsV2Request req = new
 ListObjectsV2Request().withBucketName(accessPointArn).withMaxKeys(2);
            ListObjectsV2Result result;

            do {
                result = s3Client.listObjectsV2(req);

                for (S3ObjectSummary objectSummary : result.getObjectSummaries()) {
                    System.out.printf(" - %s (size: %d)\n", objectSummary.getKey(),
 objectSummary.getSize());
                }
                // If there are more than maxKeys keys in the bucket, get a
 continuation token
                // and list the next objects.
                String token = result.getNextContinuationToken();
                System.out.println("Next Continuation Token: " + token);
                req.setContinuationToken(token);
            } while (result.isTruncated());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Menghapus objek di Amazon S3 di bucket Outposts

Objek adalah entitas dasar yang disimpan di Amazon S3 di Outposts. Setiap objek dimuat dalam bucket. Anda harus menggunakan titik akses untuk mengakses objek apapun dalam bucket Outpost. Saat Anda menentukan bucket untuk operasi objek, Anda menggunakan titik akses Amazon Resource Name (ARN) atau alias titik akses. Untuk informasi selengkapnya tentang titik akses alias, lihatMenggunakan alias untuk titik akses bucket S3 di Outposts Anda di bucket S3 di Outposts.

Contoh berikut menunjukkan format ARN untuk S3 di Outposts access points, yang mencakupWilayah AWS kode untuk Wilayah yang Outpost homed ke,Akun AWS ID, Outpost ID, dan nama access point:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Untuk informasi lebih lanjut tentang S3 di Outposts ARN, lihatARN sumber daya untuk S3 di Outposts.

Dengan Amazon S3 di Outposts, data objek selalu disimpan di Outpost. SaatAWS memasang rak Outpost, data Anda tetap lokal ke Outpost Anda untuk memenuhi persyaratan data-residensi. Objek Anda tidak pernah meninggalkan Outpost Anda dan tidak dalamWilayah AWS. Karena di-host In-region, Anda tidak dapat menggunakan konsol untuk mengunggah atau mengelola objek di Outpost Anda.AWS Management Console Namun, Anda dapat menggunakan REST API,AWS Command Line Interface (AWS CLI), danAWS SDK untuk mengunggah dan mengelola objek Anda melalui titik akses Anda.

Contoh berikut menunjukkan cara menghapus satu objek atau beberapa objek dalam bucket S3 on Outposts dengan menggunakanAWS Command Line Interface (AWS CLI) danAWS SDK for Java.

Menggunakan AWS CLI

Contoh berikut menunjukkan cara menghapus satu objek atau beberapa objek dari bucket S3 on Outposts.

delete-object

Contoh berikut menghapus sebuah objek bernama`sample-object.xml` dari S3 pada Outposts bucket (`s3-outposts:DeleteObject`) dengan menggunakanAWS CLI. Untuk menggunakan perintah ini, ganti masing-masing*`user input placeholder`* dengan informasi Anda sendiri. Untuk informasi selengkapnya tentang perintah ini, lihat delete-object di AWS CLIReferensi.

```
aws s3api delete-object --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --key sample-object.xml
```

delete-objects

Contoh berikut menghapus dua objek bernama`sample-object.xml` dan`test1.text` dari S3 pada Outposts bucket (`s3-outposts:DeleteObject`) dengan menggunakanAWS CLI. Untuk menggunakan perintah ini, ganti masing-masing*`user input placeholder`* dengan informasi Anda sendiri. Untuk informasi selengkapnya tentang perintah ini, lihat menghapus-objek di AWS CLIReferensi.

```
aws s3api delete-objects --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point --delete file://delete.json

delete.json
{
  "Objects": [
    {
      "Key": "test1.txt"
    },
    {
      "Key": "sample-object.xml"
    }
  ],
  "Quiet": false
}
```

MenggunakanAWS SDK for Java

Contoh berikut menunjukkan cara menghapus satu objek atau beberapa objek dari bucket S3 on Outposts.

DeleteObject

Contoh S3 di Outposts berikut menghapus objek dalam bucket dengan menggunakan SDK for Java. Untuk menggunakan contoh ini, tentukan titik akses ARN untuk Outpost dan nama kunci untuk objek yang ingin Anda hapus. Untuk informasi lebih lanjut, lihat DeleteObject dalam Referensi API Layanan Penyimpanan Sederhana Amazon.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
```

```
import com.amazonaws.services.s3.model.DeleteObjectRequest;

public class DeleteObject {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String keyName = "*** key name ****";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            s3Client.deleteObject(new DeleteObjectRequest(accessPointArn, keyName));
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

DeleteObjects

Contoh S3 di Outposts berikut mengunggah dan kemudian menghapus objek dalam bucket menggunakan SDK for Java. Untuk menggunakan contoh ini, tentukan titik akses ARN untuk Outpost. Untuk informasi lebih lanjut, lihat DeleteObjects dalam Referensi API Layanan Penyimpanan Sederhana Amazon.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.DeleteObjectsRequest;
import com.amazonaws.services.s3.model.DeleteObjectsRequest.KeyVersion;
import com.amazonaws.services.s3.model.DeleteObjectsResult;
```

```
import java.util.ArrayList;

public class DeleteObjects {

    public static void main(String[] args) {
       String accessPointArn = "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-
outposts-access-point";


        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Upload three sample objects.
            ArrayList<KeyVersion> keys = new ArrayList<KeyVersion>();
            for (int i = 0; i < 3; i++) {
                String keyName = "delete object example " + i;
                s3Client.putObject(accessPointArn, keyName, "Object number " + i + "
 to be deleted.");
                keys.add(new KeyVersion(keyName));
            }
            System.out.println(keys.size() + " objects successfully created.");

            // Delete the sample objects.
            DeleteObjectsRequest multiObjectDeleteRequest = new
 DeleteObjectsRequest(accessPointArn)
                    .withKeys(keys)
                    .withQuiet(false);

            // Verify that the objects were deleted successfully.
            DeleteObjectsResult delObjRes =
 s3Client.deleteObjects(multiObjectDeleteRequest);
            int successfulDeletes = delObjRes.getDeletedObjects().size();
            System.out.println(successfulDeletes + " objects successfully
 deleted.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
```

```
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Menggunakan HeadBucket untuk menentukan apakah S3 di Outposts bucket ada dan Anda memiliki izin akses

Objek adalah entitas dasar yang disimpan di Amazon S3 di Outposts. Setiap objek dimuat dalam bucket. Anda harus menggunakan titik akses untuk mengakses objek apapun di bucket Outpost. Saat Anda menentukan bucket untuk operasi objek, Anda menggunakan titik akses Amazon Resource Name (ARN) atau alias titik akses. Untuk informasi selengkapnya tentang titik akses alias, lihatMenggunakan alias untuk titik akses bucket S3 di Outposts Anda di bucket S3 di Outposts.

Contoh berikut menunjukkan format ARN untuk S3 di Outposts access points, yang mencakupWilayah AWS kode untuk Wilayah yang Outpost homed ke,Akun AWS ID, Outpost ID, dan nama access point:

```
arn:aws:s3-outposts:region:account-id:outpost/outpost-id/accesspoint/accesspoint-name
```

Untuk informasi lebih lanjut tentang S3 di Outposts ARN, lihatARN sumber daya untuk S3 di Outposts.

> **Note**
>
> Dengan Amazon S3 di Outposts, data objek selalu disimpan di Outpost. KetikaAWS menginstal rak Outpost, data Anda tetap lokal untuk Outpost Anda untuk memenuhi persyaratan data-residensi. Objek Anda tidak pernah meninggalkan Outpost Anda dan tidak dalamWilayah AWS. Karena di-host In-region, Anda tidak dapat menggunakan konsol untuk mengunggah atau mengelola objek di Outpost Anda.AWS Management Console Namun, Anda dapat menggunakan REST API,AWS Command Line Interface (AWS CLI), danAWS SDK untuk mengunggah dan mengelola objek Anda melalui titik akses Anda.

AWS Command Line Interface(AWS CLI) danAWS SDK for Java contoh berikut menunjukkan cara menggunakan operasi HeadBucket API untuk menentukan apakah bucket Amazon S3 on Outposts ada dan apakah Anda memiliki izin untuk mengaksesnya. Untuk informasi lebih lanjut, lihat HeadBucket dalam Referensi API Layanan Penyimpanan Sederhana Amazon.

Menggunakan AWS CLI

AWS CLIContoh S3 di Outposts berikut menggunakan`head-bucket` perintah untuk menentukan apakah bucket ada dan Anda memiliki izin untuk mengaksesnya. Untuk menggunakan perintah ini, ganti masing-masing*`user input placeholder`* dengan informasi Anda sendiri. Untuk informasi selengkapnya tentang perintah ini, lihat head-bucket di AWS CLIReferensi.

```
aws s3api head-bucket --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/accesspoint/example-outposts-
access-point
```

MenggunakanAWS SDK for Java

Contoh S3 di Outposts berikut menunjukkan cara menentukan apakah bucket ada dan apakah Anda memiliki izin untuk mengaksesnya. Untuk menggunakan contoh ini, tentukan titik akses ARN untuk Outpost. Untuk informasi lebih lanjut, lihat HeadBucket dalam Referensi API Layanan Penyimpanan Sederhana Amazon.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.HeadBucketRequest;

public class HeadBucket {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            s3Client.headBucket(new HeadBucketRequest(accessPointArn));
```

```
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Melakukan dan mengelola upload multipart dengan SDK for Java

Dengan Amazon S3 di Outposts, Anda dapat membuat ember S3 pada AndaAWS Outpostssumber daya dan menyimpan serta mengambil objek lokal untuk aplikasi yang memerlukan akses data lokal, pengolahan data lokal, dan residensi data. Anda dapat menggunakan S3 di Outposts melaluiAWS Management Console,AWS Command Line Interface(AWS CLI),AWSSDK, atau REST API. Untuk informasi selengkapnya, lihatApa itu Amazon S3 di Outposts?

Contoh-contoh berikut menunjukkan bagaimana Anda dapat menggunakan S3 di Outposts denganAWS SDK for Javauntuk melakukan dan mengelola unggahan multipart.

Topik

- Melakukan unggahan multipart sebuah objek di bucket S3 di Outposts
- Salin objek besar di bucket S3 di Outposts dengan menggunakan unggahan multipart
- Buat daftar bagian objek di bucket S3 di Outposts
- Dapatkan daftar unggahan multibagian yang sedang berlangsung di bucket S3 di Outposts

Melakukan unggahan multipart sebuah objek di bucket S3 di Outposts

Contoh S3 di Outposts berikut memulai, mengunggah, dan menyelesaikan unggahan multibagian objek ke bucket dengan menggunakan SDK for Java. Untuk menggunakan contoh ini, ganti masing-masing*user input placeholder*dengan informasi Anda sendiri. Untuk informasi selengkapnya, lihat Pengunggahan objek menggunakan unggahan multibagian.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
```

```
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.ArrayList;
import java.util.List;

public class MultipartUploadCopy {
    public static void main(String[] args) {
        String accessPointArn = "*** Source access point ARN ***";
        String sourceObjectKey = "*** Source object key ***";
        String destObjectKey = "*** Target object key ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
 InitiateMultipartUploadRequest(accessPointArn, destObjectKey);
            InitiateMultipartUploadResult initResult =
 s3Client.initiateMultipartUpload(initRequest);

            // Get the object size to track the end of the copy operation.
            GetObjectMetadataRequest metadataRequest = new
 GetObjectMetadataRequest(accessPointArn, sourceObjectKey);
            ObjectMetadata metadataResult =
 s3Client.getObjectMetadata(metadataRequest);
            long objectSize = metadataResult.getContentLength();

            // Copy the object using 5 MB parts.
            long partSize = 5 * 1024 * 1024;
            long bytePosition = 0;
            int partNum = 1;
            List<CopyPartResult> copyResponses = new ArrayList<CopyPartResult>();
            while (bytePosition < objectSize) {
                // The last part might be smaller than partSize, so check to make sure
                // that lastByte isn't beyond the end of the object.
                long lastByte = Math.min(bytePosition + partSize - 1, objectSize - 1);

                // Copy this part.
```

```
                CopyPartRequest copyRequest = new CopyPartRequest()
                        .withSourceBucketName(accessPointArn)
                        .withSourceKey(sourceObjectKey)
                        .withDestinationBucketName(accessPointArn)
                        .withDestinationKey(destObjectKey)
                        .withUploadId(initResult.getUploadId())
                        .withFirstByte(bytePosition)
                        .withLastByte(lastByte)
                        .withPartNumber(partNum++);
                copyResponses.add(s3Client.copyPart(copyRequest));
                bytePosition += partSize;
            }

            // Complete the upload request to concatenate all uploaded parts and make
the copied object available.
            CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest(
                    accessPointArn,
                    destObjectKey,
                    initResult.getUploadId(),
                    getETags(copyResponses));
            s3Client.completeMultipartUpload(completeRequest);
            System.out.println("Multipart copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }

    // This is a helper function to construct a list of ETags.
    private static List<PartETag> getETags(List<CopyPartResult> responses) {
        List<PartETag> etags = new ArrayList<PartETag>();
        for (CopyPartResult response : responses) {
            etags.add(new PartETag(response.getPartNumber(), response.getETag()));
        }
        return etags;
    }
```

Salin objek besar di bucket S3 di Outposts dengan menggunakan unggahan multipart

Contoh S3 di Outposts berikut menggunakan SDK for Java untuk menyalin objek dalam bucket. Untuk menggunakan contoh ini, ganti masing-masing*user input placeholder*dengan informasi Anda sendiri. Contoh ini diadaptasi dari[Menyalin objek menggunakan unggahan multibagian](#).

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.ArrayList;
import java.util.List;

public class MultipartUploadCopy {
    public static void main(String[] args) {
        String accessPointArn = "*** Source access point ARN ***";
        String sourceObjectKey = "*** Source object key ***";
        String destObjectKey = "*** Target object key ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Initiate the multipart upload.
            InitiateMultipartUploadRequest initRequest = new
 InitiateMultipartUploadRequest(accessPointArn, destObjectKey);
            InitiateMultipartUploadResult initResult =
 s3Client.initiateMultipartUpload(initRequest);

            // Get the object size to track the end of the copy operation.
            GetObjectMetadataRequest metadataRequest = new
 GetObjectMetadataRequest(accessPointArn, sourceObjectKey);
            ObjectMetadata metadataResult =
 s3Client.getObjectMetadata(metadataRequest);
            long objectSize = metadataResult.getContentLength();

            // Copy the object using 5 MB parts.
```

```
            long partSize = 5 * 1024 * 1024;
            long bytePosition = 0;
            int partNum = 1;
            List<CopyPartResult> copyResponses = new ArrayList<CopyPartResult>();
            while (bytePosition < objectSize) {
                // The last part might be smaller than partSize, so check to make sure
                // that lastByte isn't beyond the end of the object.
                long lastByte = Math.min(bytePosition + partSize - 1, objectSize - 1);

                // Copy this part.
                CopyPartRequest copyRequest = new CopyPartRequest()
                        .withSourceBucketName(accessPointArn)
                        .withSourceKey(sourceObjectKey)
                        .withDestinationBucketName(accessPointArn)
                        .withDestinationKey(destObjectKey)
                        .withUploadId(initResult.getUploadId())
                        .withFirstByte(bytePosition)
                        .withLastByte(lastByte)
                        .withPartNumber(partNum++);
                copyResponses.add(s3Client.copyPart(copyRequest));
                bytePosition += partSize;
            }

            // Complete the upload request to concatenate all uploaded parts and make
the copied object available.
            CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest(
                    accessPointArn,
                    destObjectKey,
                    initResult.getUploadId(),
                    getETags(copyResponses));
            s3Client.completeMultipartUpload(completeRequest);
            System.out.println("Multipart copy complete.");
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
```

```
    // This is a helper function to construct a list of ETags.
    private static List<PartETag> getETags(List<CopyPartResult> responses) {
        List<PartETag> etags = new ArrayList<PartETag>();
        for (CopyPartResult response : responses) {
            etags.add(new PartETag(response.getPartNumber(), response.getETag()));
        }
        return etags;
    }
}
```

Buat daftar bagian objek di bucket S3 di Outposts

Contoh S3 di Outposts berikut mencantumkan bagian objek di bucket dengan menggunakan SDK for Java. Untuk menggunakan contoh ini, ganti masing-masing*user input placeholder*dengan informasi Anda sendiri.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;

import java.util.List;

public class ListParts {
    public static void main(String[] args) {
        String accessPointArn = "*** access point ARN ***";
        String keyName = "*** Key name ***";
        String uploadId = "*** Upload ID ***";

        try {
            // This code expects that you have AWS credentials set up per:
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            ListPartsRequest listPartsRequest = new ListPartsRequest(accessPointArn,
 keyName, uploadId);
            PartListing partListing = s3Client.listParts(listPartsRequest);
            List<PartSummary> partSummaries = partListing.getParts();
```

```
            System.out.println(partSummaries.size() + " multipart upload parts");
            for (PartSummary p : partSummaries) {
                System.out.println("Upload part: Part number = \"" + p.getPartNumber()
 + "\", ETag = " + p.getETag());
            }

        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Dapatkan daftar unggahan multibagian yang sedang berlangsung di bucket S3 di Outposts

Contoh S3 di Outposts berikut memperlihatkan cara mengambil daftar unggahan multibagian yang sedang berlangsung dengan bucket SDK for Java. Untuk menggunakan contoh ini, ganti masing-masing*user input placeholder*dengan informasi Anda sendiri. Contoh ini diadaptasi dari[Mendaftarkan unggahan multibagian](#)contoh untuk Amazon S3.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.ListMultipartUploadsRequest;
import com.amazonaws.services.s3.model.MultipartUpload;
import com.amazonaws.services.s3.model.MultipartUploadListing;

import java.util.List;

public class ListMultipartUploads {
    public static void main(String[] args) {
                String accessPointArn = "*** access point ARN ***";

        try {
            // This code expects that you have AWS credentials set up per:
```

```
            // https://docs.aws.amazon.com/sdk-for-java/v1/developer-guide/setup-
credentials.html
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .enableUseArnRegion()
                    .build();

            // Retrieve a list of all in-progress multipart uploads.
            ListMultipartUploadsRequest allMultipartUploadsRequest = new
 ListMultipartUploadsRequest(accessPointArn);
            MultipartUploadListing multipartUploadListing =
 s3Client.listMultipartUploads(allMultipartUploadsRequest);
            List<MultipartUpload> uploads =
 multipartUploadListing.getMultipartUploads();

            // Display information about all in-progress multipart uploads.
            System.out.println(uploads.size() + " multipart upload(s) in progress.");
            for (MultipartUpload u : uploads) {
                System.out.println("Upload in progress: Key = \"" + u.getKey() + "\",
 id = " + u.getUploadId());
            }
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

Menggunakan URL yang telah ditandatangani untuk S3 di Outposts

Untuk memberikan akses terbatas waktu ke objek yang disimpan secara lokal di Outpost tanpa memperbarui kebijakan bucket, Anda dapat menggunakan URL yang telah ditandatangani sebelumnya. Dengan URL yang telah ditandatangani sebelumnya, Anda sebagai pemilik bucket dapat berbagi objek dengan individu di cloud pribadi virtual (VPC) Anda atau memberi mereka kemampuan untuk mengunggah atau menghapus objek.

Saat Anda menciptakan sebuah URL presigned dengan menggunakanAWS URL atauAWS Command Line Interface (AWS CLI), Anda mengaitkan URL dengan sebuah tindakan tertentu. Anda

juga memberikan akses terbatas waktu ke URL yang telah ditandatangani sebelumnya dengan memilih waktu kedaluwarsa kustom yang bisa serendah 1 detik dan setinggi 7 hari. Saat Anda membagikan URL presigned, individu di Outposts dapat melakukan tindakan yang disertakan dalam URL seolah-olah mereka adalah pengguna yang memublikasikan sendiri. Ketika URL mencapai waktu kedaluwarsa, URL kedaluwarsa dan tidak lagi berfungsi.

Pembatasan kemampuan URL presigned

Kemampuan URL presigned dibatasi oleh izin pengguna yang membuatnya. Pada intinya, URL presigned adalah token pembawa yang memberikan akses kepada mereka yang memilikinya. Oleh karena itu, kami menyarankan agar Anda melindunginya sebagaimana mestinya.

AWSTanda tangan Versi 4 (SigV4)

Untuk menerapkan perilaku tertentu saat permintaan URL yang ditandatangani sebelumnya diautentikasi dengan menggunakanAWS Signature Version 4 (SiGv4), Anda dapat menggunakan kunci kondisi dalam kebijakan bucket dan kebijakan titik akses. Misalnya, Anda dapat membuat kebijakan bucket yang menggunakan`s3-outposts:signatureAge` kondisi tersebut untuk menolak permintaan URL Amazon S3 pada Outposts yang ditandatangani sebelumnya pada objek di`example-outpost-bucket` bucket jika tanda tangan berusia lebih dari 10 menit. Untuk menggunakan contoh ini, ganti *`user input placeholders`*dengan informasi Anda sendiri.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Deny a presigned URL request if the signature is more than 10
 minutes old",
            "Effect": "Deny",
            "Principal": {"AWS":"444455556666"},
            "Action": "s3-outposts:*",
            "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/object/
*",
            "Condition": {
                "NumericGreaterThan": {"s3-outposts:signatureAge": 600000},
                "StringEquals": {"s3-outposts:authType": "REST-QUERY-STRING"}
            }
        }
    ]
}
```

Untuk daftar kunci kondisi dan kebijakan contoh tambahan yang dapat Anda gunakan untuk menegakkan perilaku tertentu saat permintaan URL yang ditandatangani sebelumnya diautentikasi menggunakan Tanda Tangan Versi 4, lihatAWSKunci kebijakan autentikasi Versi 4 (SiGv4).

Pembatasan jalur jaringan

Jika Anda ingin membatasi penggunaan URL presigned dan semua S3 di Outposts akses ke jalur jaringan tertentu, Anda dapat tulis kebijakan yang memerlukan jalur jaringan tertentu. Untuk menetapkan pembatasan pada prinsipal IAM yang melakukan panggilan, Anda dapat menggunakan kebijakan berbasis identitasAWS Identity and Access Management (IAM) (misalnya, kebijakan pengguna, grup, atau peran). Untuk mengatur pembatasan sumber daya S3 on Outposts, Anda dapat menggunakan kebijakan berbasis sumber daya (misalnya, kebijakan bucket dan access point).

Pembatasan jalur jaringan kepada penanggung jawab mengharuskan pengguna kredensyal tersebut membuat permintaan dari jaringan tertentu. Sebuah pembatasan pada bucket atau titik akses mengharuskan semua permintaan ke sumber daya tersebut berasal dari jaringan tertentu. Pembatasan ini juga berlaku di Outposts URL presigned.

Kondisi global IAM yang Anda gunakan bergantung pada jenis titik akhir. Jika Anda menggunakan titik akhir publik untuk S3 di Outposts, gunakan`aws:SourceIp`. Jika Anda menggunakan endpoint VPC untuk S3 di Outposts, gunakan`aws:SourceVpc` atau`aws:SourceVpce`.

Pernyataan kebijakan IAM berikut mengharuskan penanggung jawab untuk mengakses dari rentang jaringan tertentuAWS saja. Dengan pernyataan kebijakan ini, semua akses harus berasal dari rentang itu. Kebijakan ini juga termasuk kasus seseorang yang menggunakan URL presigned untuk S3 di Outposts. Untuk menggunakan contoh ini, ganti *`user input placeholders`*dengan informasi Anda sendiri.

```
{
    "Sid": "NetworkRestrictionForIAMPrincipal",
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
        "NotIpAddressIfExists": {"aws:SourceIp": "IP-address-range"},
        "BoolIfExists": {"aws:ViaAWSService": "false"}
    }
}
```

Untuk contoh kebijakan bucket yang menggunakan kunci kondisi`aws:SourceIP`AWS global untuk membatasi akses ke bucket S3 di Outposts ke kisaran jaringan tertentu, lihatMengatur IAM dengan S3 di Outposts.

Siapa yang dapat menciptakan URL presigned

Siapa pun yang memiliki kredensial keamanan yang valid dapat menciptakan sebuah URL presigned. Akan tetapi agar pengguna di VPC berhasil mengakses sebuah objek, URL presigned harus diciptakan oleh seseorang yang memiliki izin untuk melakukan operasi yang menjadi dasar URL presigned tersebut.

Anda dapat menggunakan kredensyal berikut untuk membuat URL presigned:

- Profil instans IAM - Valid hingga 6 jam.
- AWS Security Token Service- Berlaku hingga 36 jam saat ditandatangani dengan kredensyal permanen, seperti kredensyal penggunaAkun AWS root atau pengguna IAM.
- Pengguna IAM berlaku hingga 7 hari saat Anda menggunakanAWS Tanda tangan Versi 4.

  Untuk menciptakan sebuah URL presigned yang berlaku hingga 7 hari, pertama-tama tentukan kredensyal pengguna IAM (kunci akses dan kunci rahasia) pada SDK yang Anda gunakan. Kemudian, buat URL presigned dengan menggunakanAWS Tanda tangan Versi 4.

> **Note**
>
> - Jika Anda menciptakan sebuah URL presigned menggunakan token sementara, URL akan kedaluwarsa saat token kedaluwarsa, meskipun Anda membuat URL dengan waktu kedaluwarsa lebih lama.
> - Karena URL presigned memberikan akses ke bucket S3 di Outposts Anda kepada siapa pun yang memiliki URL tersebut, kami merekomendasikan agar Anda melindunginya dengan tepat. Untuk informasi selengkapnya tentang perlindungan URL presigned, lihatPembatasan kemampuan URL presigned.

Kapan S3 di Outposts memeriksa tanggal dan waktu kedaluwarsa URL presigned?

Pada saat permintaan HTTP, S3 di Outposts memeriksa tanggal kedaluwarsa dan waktu URL yang ditandatangani. Misalnya, jika klien mulai mengunduh file besar segera sebelum waktu kedaluwarsa,

pengunduhan berlanjut meskipun waktu kedaluwarsa sudah lewat selama pengunduhan. Akan tetapi, jika koneksi menurun dan klien mencoba memulai ulang unduhan setelah waktu kedaluwarsa berlalu, pengunduhan gagal.

Untuk informasi selengkapnya tentang menggunakan URL presigned untuk berbagi atau unggah objek, lihat topik berikut.

Topik

- Berbagi objek dengan menggunakan URL yang telah ditandatangani
- Membuat sebuah URL presigned untuk mengunggah sebuah objek ke bucket S3 di Outposts

Berbagi objek dengan menggunakan URL yang telah ditandatangani

Untuk memberikan akses terbatas waktu ke objek yang disimpan secara lokal di Outpost tanpa memperbarui kebijakan bucket, Anda dapat menggunakan URL yang telah ditetapkan sebelumnya. Dengan URL yang telah ditetapkan sebelumnya, Anda sebagai pemilik bucket dapat berbagi objek dengan individu di virtual private cloud (VPC) Anda atau memberi mereka kemampuan untuk mengunggah atau menghapus objek.

Saat membuat URL presigned menggunakan AWS SDK atau AWS Command Line Interface (AWS CLI), Anda mengaitkan URL dengan tindakan tertentu. Anda juga memberikan akses terbatas waktu ke URL yang telah ditetapkan sebelumnya dengan memilih waktu kedaluwarsa kustom yang bisa serendah 1 detik dan setinggi 7 hari. Saat Anda membagikan URL yang telah ditandatangani, individu di VPC dapat melakukan tindakan yang disertakan dalam URL seolah-olah mereka adalah pengguna yang memublikasikan sendiri. Ketika URL mencapai waktu kedaluwarsa, URL kedaluwarsa dan tidak lagi berfungsi.

Saat Anda membuat sebuah URL yang telah ditandatangani, Anda harus memberikan kredensial keamanan Anda, lalu menentukan berikut ini:

- Titik akses Amazon Resource Name (ARN) untuk Amazon S3 di bucket Outposts
- Kunci objek
- Metode HTTP (`GET`untuk mengunduh objek)
- Tanggal dan waktu kedaluwarsa

Sebuah URL presigned hanya berlaku selama durasi yang telah ditentukan. Artinya, Anda harus memulai tindakan yang diizinkan oleh URL sebelum tanggal dan waktu kedaluwarsa. Anda dapat

menggunakan sebuah URL presigned berkali-kali, hingga tanggal dan waktu kedaluwarsa. Jika Anda membuat URL yang telah ditandatangani menggunakan token sementara, maka URL akan kedaluwarsa saat token kedaluwarsa, meskipun jika Anda membuat URL dengan waktu kedaluwarsa yang lebih lama.

Pengguna di virtual private cloud (VPC) yang memiliki akses ke URL presigned dapat mengakses objek. Misalnya, jika Anda memiliki video dalam bucket Anda dan bucket maupun objek tersebut bersifat pribadi, Anda dapat membagikan video dengan orang lain dengan membuat URL presigned. Karena URL yang telah ditandatangani memberikan akses ke bucket S3 on Outposts Anda kepada siapa pun yang memiliki URL tersebut, kami merekomendasikan agar Anda melindungi URL ini dengan tepat. Untuk detail lebih lanjut tentang perlindungan URL yang telah ditandatangani, lihat Pembatasan kemampuan URL presigned.

Siapa pun yang memiliki kredensial keamanan yang valid dapat membuat sebuah URL yang telah ditandatangani. Akan tetapi, URL presigned harus dibuat oleh seseorang yang memiliki izin untuk melakukan operasi yang menjadi dasar URL presigned. Untuk informasi selengkapnya, lihat Siapa yang dapat menciptakan URL presigned.

Anda dapat membuat URL yang telah ditetapkan sebelumnya untuk berbagi objek di bucket S3 di Outposts dengan menggunakan SDK AWS dan AWS CLI. Untuk informasi selengkapnya, lihat contoh berikut ini.

Menggunakan AWS SDK

Anda dapat menggunakan AWS SDK untuk menghasilkan URL presigned yang dapat Anda berikan kepada orang lain sehingga mereka dapat mengambil objek.

> **Note**
>
> Saat Anda menggunakan AWS SDK untuk membuat URL yang telah ditetapkan sebelumnya, waktu kedaluwarsa maksimum untuk URL yang telah ditetapkan sebelumnya adalah 7 hari dari waktu pembuatan.

Java

Example

Contoh berikut menampilkan pembuatan URL presigned yang dapat Anda berikan kepada orang lain sehingga mereka dapat mengambil objek dari bucket S3 di Outposts. Untuk informasi

selengkapnya, lihat Menggunakan URL yang telah ditandatangani untuk S3 di Outposts. Untuk menggunakan contoh ini, ganti *user input placeholders*dengan informasi Anda sendiri.

Untuk petunjuk cara membuat dan menguji sampel yang berfungsi, lihat Memulai di Panduan AWS SDK for Java Pengembang.

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.HttpMethod;
import com.amazonaws.SdkClientException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.GeneratePresignedUrlRequest;

import java.io.IOException;
import java.net.URL;
import java.time.Instant;

public class GeneratePresignedURL {

    public static void main(String[] args) throws IOException {
        Regions clientRegion = Regions.DEFAULT_REGION;
        String accessPointArn = "*** access point ARN ***";
        String objectKey = "*** object key ***";

        try {
            AmazonS3 s3Client = AmazonS3ClientBuilder.standard()
                    .withRegion(clientRegion)
                    .withCredentials(new ProfileCredentialsProvider())
                    .build();

            // Set the presigned URL to expire after one hour.
            java.util.Date expiration = new java.util.Date();
            long expTimeMillis = Instant.now().toEpochMilli();
            expTimeMillis += 1000 * 60 * 60;
            expiration.setTime(expTimeMillis);

            // Generate the presigned URL.
            System.out.println("Generating pre-signed URL.");
            GeneratePresignedUrlRequest generatePresignedUrlRequest =
                    new GeneratePresignedUrlRequest(accessPointArn, objectKey)
                            .withMethod(HttpMethod.GET)
```

```
                                .withExpiration(expiration);
            URL url = s3Client.generatePresignedUrl(generatePresignedUrlRequest);

            System.out.println("Pre-Signed URL: " + url.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 couldn't
 process
            // it, so it returned an error response.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 couldn't be contacted for a response, or the client
            // couldn't parse the response from Amazon S3.
            e.printStackTrace();
        }
    }
}
```

.NET

Example

Contoh berikut menampilkan pembuatan URL presigned yang dapat Anda berikan kepada orang lain sehingga mereka dapat mengambil objek dari bucket S3 di Outposts. Untuk informasi selengkapnya, lihat Menggunakan URL yang telah ditandatangani untuk S3 di Outposts. Untuk menggunakan contoh ini, ganti *user input placeholders*dengan informasi Anda sendiri.

Untuk informasi tentang menyiapkan dan menjalankan contoh kode, lihat Memulai SDK for .NET di AWSAWS SDK for .NET Developer Guide.

```
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using System;

namespace Amazon.DocSamples.S3
{
    class GenPresignedURLTest
    {
        private const string accessPointArn = "*** access point ARN ***";
        private const string objectKey = "*** object key ***";
        // Specify how long the presigned URL lasts, in hours.
        private const double timeoutDuration = 12;
        // Specify your bucket Region (an example Region is shown).
```

```
        private static readonly RegionEndpoint bucketRegion =
 RegionEndpoint.USWest2;
        private static IAmazonS3 s3Client;

        public static void Main()
        {
            s3Client = new AmazonS3Client(bucketRegion);
            string urlString = GeneratePreSignedURL(timeoutDuration);
        }
        static string GeneratePreSignedURL(double duration)
        {
            string urlString = "";
            try
            {
                GetPreSignedUrlRequest request1 = new GetPreSignedUrlRequest
                {
                    BucketName = accessPointArn,
                    Key = objectKey,
                    Expires = DateTime.UtcNow.AddHours(duration)
                };
                urlString = s3Client.GetPreSignedURL(request1);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine("Error encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            catch (Exception e)
            {
                Console.WriteLine("Unknown encountered on server. Message:'{0}' when
 writing an object", e.Message);
            }
            return urlString;
        }
    }
}
```

Python

Contoh berikut menampilkan pembuatan URL presigned untuk berbagi objek dengan menggunakan SDK untuk Python (Boto3). Sebagai contoh, gunakan klien Boto3 dan `generate_presigned_url` fungsi untuk menghasilkan URL presigned yang memungkinkan Anda untuk `GET` objek.

```
import boto3
    url = boto3.client('s3').generate_presigned_url(
    ClientMethod='get_object',
    Params={'Bucket': 'ACCESS_POINT_ARN', 'Key': 'OBJECT_KEY'},
    ExpiresIn=3600)
```

Untuk informasi lebih lanjut tentang penggunaan SDK untuk Python (Boto3) untuk menghasilkan URL yang telah ditandatangani sebelumnya, lihat Python di Referensi API AWS SDK for Python (Boto) .

Menggunakan AWS CLI

AWS CLI Perintah contoh berikut menghasilkan URL presigned untuk bucket S3 pada Outposts. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

> **Note**
>
> Saat Anda menggunakan AWS CLI untuk membuat URL yang telah ditetapkan sebelumnya, waktu kedaluwarsa maksimum untuk URL yang telah ditetapkan sebelumnya adalah 7 hari dari waktu pembuatan.

```
aws s3 presign s3://arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/example-outpost-access-
point/mydoc.txt --expires-in 604800
```

Untuk informasi selengkapnya, lihat presign dalam Referensi AWS CLI Perintah.

Membuat sebuah URL presigned untuk mengunggah sebuah objek ke bucket S3 di Outposts

Untuk memberikan akses terbatas waktu ke objek yang disimpan secara lokal di Outpost tanpa memperbarui kebijakan bucket, Anda dapat menggunakan URL yang telah ditandatangani sebelumnya. Dengan URL yang telah ditandatangani sebelumnya, Anda sebagai pemilik bucket dapat berbagi objek dengan individu di cloud pribadi virtual (VPC) Anda atau memberi mereka kemampuan untuk mengunggah atau menghapus objek.

Saat Anda menciptakan sebuah URL presigned dengan menggunakanAWSSDK atau SDKAWS Command Line Interface(AWS CLI), Anda mengaitkan URL dengan sebuah tindakan tertentu. Anda

juga memberikan akses terbatas waktu ke URL yang telah ditandatangani sebelumnya dengan memilih waktu kedaluwarsa kustom yang bisa serendah 1 detik dan setinggi 7 hari. Saat Anda membagikan URL presigned, individu di VPC dapat melakukan tindakan yang disertakan dalam URL seolah-olah mereka adalah pengguna yang membolikasikannya sendiri. Ketika URL mencapai waktu kedaluwarsa, URL kedaluwarsa dan tidak lagi berfungsi.

Saat Anda menciptakan sebuah URL presigned, Anda harus memberikan kredensi keamanan Anda, dan kemudian menentukan yang berikut:

- Titik akses Amazon Resource Name (ARN) untuk bucket Amazon S3 on Outposts
- Kunci objek
- Metode HTTP (`PUT`untuk mengunggah objek)
- Tanggal dan waktu kedaluwarsa

Sebuah URL presigned hanya berlaku selama durasi yang telah ditentukan. Artinya, Anda harus memulai tindakan yang diizinkan oleh URL sebelum tanggal dan waktu kedaluwarsa. Anda dapat menggunakan sebuah URL presigned berkali-kali, hingga tanggal dan waktu kedaluwarsa. Jika Anda menciptakan sebuah URL presigned dengan menggunakan token sementara, maka URL akan kedaluwarsa saat token kedaluwarsa, meskipun Anda menciptakan sebuah URL dengan waktu kedaluwarsa lebih lama.

Jika tindakan yang diizinkan oleh sebuah URL presigned terdiri dari beberapa langkah, seperti unggahan multipart, Anda harus memulai semua langkah sebelum waktu kedaluwarsa. Jika S3 di Outposts mencoba memulai langkah dengan URL yang kedaluwarsa, Anda menerima kesalahan.

Pengguna di virtual private cloud (VPC) yang memiliki akses ke URL presigned dapat mengunggah objek. Misalnya, pengguna di VPC yang memiliki akses ke URL yang telah ditandatangani sebelumnya dapat mengunggah objek ke bucket Anda. Karena URL presigned memberikan akses ke bucket S3 Anda di Outposts ke setiap pengguna di VPC yang memiliki akses ke URL presigned, kami merekomendasikan agar Anda melindungi URL ini dengan tepat. Untuk detail lebih lanjut tentang perlindungan URL yang telah ditandatangani, lihat Pembatasan kemampuan URL presigned.

Siapa pun yang memiliki kredensial keamanan yang valid dapat menciptakan sebuah URL presigned. Akan tetapi, URL presigned harus diciptakan oleh seseorang yang memiliki izin untuk melakukan operasi yang menjadi dasar URL presigned. Untuk informasi selengkapnya, lihat Siapa yang dapat menciptakan URL presigned.

MenggunakanAWSSDK untuk Java guna menghasilkan sebuah URL presigned untuk operasi objek S3 on Outposts

Java

SDK for Java 2.x

Contoh ini menunjukkan cara menghasilkan sebuah URL presigned yang dapat digunakan untuk mengunggah sebuah objek ke sebuah bucket S3 di Outposts selama waktu yang terbatas. Untuk informasi selengkapnya, lihat [Menggunakan URL yang telah ditandatangani untuk S3 di Outposts](#).

```
   public static void signBucket(S3Presigner presigner, String
outpostAccessPointArn, String keyName) {

       try {
           PutObjectRequest objectRequest = PutObjectRequest.builder()
                   .bucket(accessPointArn)
                   .key(keyName)
                   .contentType("text/plain")
                   .build();

           PutObjectPresignRequest presignRequest =
PutObjectPresignRequest.builder()
                   .signatureDuration(Duration.ofMinutes(10))
                   .putObjectRequest(objectRequest)
                   .build();

           PresignedPutObjectRequest presignedRequest =
presigner.presignPutObject(presignRequest);


           String myURL = presignedRequest.url().toString();
           System.out.println("Presigned URL to upload a file to: " +myURL);
           System.out.println("Which HTTP method must be used when uploading a
file: " +
                   presignedRequest.httpRequest().method());

           // Upload content to the S3 on Outposts bucket by using this URL.
           URL url = presignedRequest.url();

           // Create the connection and use it to upload the new object by using
the presigned URL.
```

```
            HttpURLConnection connection = (HttpURLConnection)
 url.openConnection();
            connection.setDoOutput(true);
            connection.setRequestProperty("Content-Type","text/plain");
            connection.setRequestMethod("PUT");
            OutputStreamWriter out = new
 OutputStreamWriter(connection.getOutputStream());
            out.write("This text was uploaded as an object by using a presigned
 URL.");
            out.close();

            connection.getResponseCode();
            System.out.println("HTTP response code is " +
 connection.getResponseCode());

        } catch (S3Exception e) {
            e.getStackTrace();
        } catch (IOException e) {
            e.getStackTrace();
        }
    }
```

Python

SDK for Python (Boto3)

Contoh ini menunjukkan cara membuat URL yang telah ditandatangani sebelumnya yang dapat melakukan aksi S3 on Outposts untuk waktu yang terbatas. Untuk informasi selengkapnya, lihat Menggunakan URL yang telah ditandatangani untuk S3 di Outposts. Untuk membuat permintaan dengan URL, gunakan`Requests`paket.

```
import argparse
import logging
import boto3
from botocore.exceptions import ClientError
import requests

logger = logging.getLogger(__name__)


def generate_presigned_url(s3_client, client_method, method_parameters,
 expires_in):
```

```
    """
    Generate a presigned S3 on Outposts URL that can be used to perform an
 action.

    :param s3_client: A Boto3 Amazon S3 client.
    :param client_method: The name of the client method that the URL performs.
    :param method_parameters: The parameters of the specified client method.
    :param expires_in: The number of seconds that the presigned URL is valid for.
    :return: The presigned URL.
    """
    try:
        url = s3_client.generate_presigned_url(
            ClientMethod=client_method,
            Params=method_parameters,
            ExpiresIn=expires_in
        )
        logger.info("Got presigned URL: %s", url)
    except ClientError:
        logger.exception(
            "Couldn't get a presigned URL for client method '%s'.",
 client_method)
        raise
    return url


def usage_demo():
    logging.basicConfig(level=logging.INFO, format='%(levelname)s: %(message)s')

    print('-'*88)
    print("Welcome to the Amazon S3 on Outposts presigned URL demo.")
    print('-'*88)

    parser = argparse.ArgumentParser()
    parser.add_argument('accessPointArn', help="The name of the S3 on Outposts
 access point ARN.")
    parser.add_argument(
        'key', help="For a GET operation, the key of the object in S3 on
 Outposts. For a "
                    "PUT operation, the name of a file to upload.")
    parser.add_argument(
        'action', choices=('get', 'put'), help="The action to perform.")
    args = parser.parse_args()

    s3_client = boto3.client('s3')
```

```
    client_action = 'get_object' if args.action == 'get' else 'put_object'
    url = generate_presigned_url(
        s3_client, client_action, {'Bucket': args.accessPointArn, 'Key':
 args.key}, 1000)

    print("Using the Requests package to send a request to the URL.")
    response = None
    if args.action == 'get':
        response = requests.get(url)
    elif args.action == 'put':
        print("Putting data to the URL.")
        try:
            with open(args.key, 'r') as object_file:
                object_text = object_file.read()
            response = requests.put(url, data=object_text)
        except FileNotFoundError:
            print(f"Couldn't find {args.key}. For a PUT operation, the key must
 be the "
                  f"name of a file that exists on your computer.")

    if response is not None:
        print("Got response:")
        print(f"Status: {response.status_code}")
        print(response.text)

    print('-'*88)


if __name__ == '__main__':
    usage_demo()
```

Amazon S3 di Outposts dengan Amazon EMR lokal di Outposts

Amazon EMR adalah platform cluster terkelola yang menyederhanakan menjalankan kerangka kerja data besar, seperti Apache Hadoop danApache Spark, AWS untuk memproses dan menganalisis sejumlah besar data. Dengan menggunakan kerangka kerja ini dan proyek sumber terbuka terkait, Anda dapat memproses data untuk tujuan analitik dan beban kerja intelijen bisnis. Amazon EMR juga membantu Anda mengubah dan memindahkan sejumlah besar data ke dalam dan keluar dari penyimpanan AWS data dan database lainnya, serta mendukung Amazon S3 di Outposts. Untuk informasi selengkapnya tentang Amazon EMR, lihat Amazon EMR di Outposts di Panduan Manajemen EMR Amazon.

Untuk Amazon S3 di Outposts, Amazon EMR mulai mendukung konektor S3A di versi 7.0.0Apache Hadoop. Versi Amazon EMR sebelumnya tidak mendukung S3 lokal di Outposts, dan EMR File System (EMRFS) tidak didukung.

Aplikasi-aplikasi yang didukung

Amazon EMR dengan Amazon S3 di Outposts mendukung aplikasi berikut:

- Hadoop
- Spark
- Hue
- Hive
- Sqoop
- Pig
- Hudi
- Flink

Untuk informasi selengkapnya, lihat Panduan Rilis Amazon EMR.

Membuat dan mengonfigurasi bucket Amazon S3 di Outposts

Amazon EMR menggunakan AWS SDK for Java dengan Amazon S3 di Outposts untuk menyimpan data input dan data output. File log EMR Amazon Anda disimpan di lokasi Amazon S3 Regional yang Anda pilih dan tidak disimpan secara lokal di Outpost. Untuk informasi selengkapnya, lihat log EMR Amazon di Panduan Manajemen EMR Amazon.

Agar sesuai dengan persyaratan Amazon S3 dan DNS, bucket S3 on Outposts memiliki batasan dan batasan penamaan. Untuk informasi selengkapnya, lihat Membuat bucket S3 di Outposts.

Dengan Amazon EMR versi 7.0.0 dan yang lebih baru, Anda dapat menggunakan Amazon EMR dengan S3 di Outposts dan sistem file S3A.

Prasyarat

Izin S3 on Outposts — Saat Anda membuat profil instans EMR Amazon, peran Anda harus berisi namespace (IAM) untuk S3 AWS Identity and Access Management di Outposts. S3 di Outposts memiliki namespace sendiri,. `s3-outposts*` Untuk contoh kebijakan yang menggunakan namespace ini, lihat. Mengatur IAM dengan S3 di Outposts

Konektor S3A - Untuk mengonfigurasi kluster EMR Anda untuk mengakses data dari bucket Amazon S3 pada Outposts, Anda harus menggunakan konektor S3A. Apache Hadoop Untuk menggunakan konektor, pastikan bahwa semua URI S3 Anda menggunakan skema. `s3a` Jika tidak, Anda dapat mengonfigurasi implementasi sistem file yang Anda gunakan untuk cluster EMR Anda sehingga URI S3 Anda bekerja dengan konektor S3A.

Untuk mengonfigurasi implementasi sistem file agar berfungsi dengan konektor S3A, Anda menggunakan properti `fs.`*`file_scheme`*`.impl` dan `fs.AbstractFileSystem.`*`file_scheme`*`.impl` konfigurasi untuk cluster EMR Anda, yang *`file_scheme`* sesuai dengan jenis URI S3 yang Anda miliki. Untuk menggunakan contoh berikut, ganti *`user input placeholders`* dengan informasi Anda sendiri. Misalnya, untuk mengubah implementasi sistem file untuk URI S3 yang menggunakan `s3` skema, tentukan properti konfigurasi cluster berikut:

```
[
  {
"Classification": "core-site",
    "Properties": {
    "fs.s3.impl": "org.apache.hadoop.fs.s3a.S3AFileSystem",
    "fs.AbstractFileSystem.s3.impl": "org.apache.hadoop.fs.s3a.S3A"
    }
  }
]
```

Untuk menggunakan S3A, atur properti `fs.`*`file_scheme`*`.impl` konfigurasi ke`org.apache.hadoop.fs.s3a.S3AFileSystem`, dan setel `fs.AbstractFileSystem.`*`file_scheme`*`.impl` properti ke. `org.apache.hadoop.fs.s3a.S3A`

Misalnya, jika Anda mengakses jalur`s3a://`*`bucket`*`/...`, atur `fs.s3a.impl` properti ke`org.apache.hadoop.fs.s3a.S3AFileSystem`, dan setel `fs.AbstractFileSystem.s3a.impl` properti ke`org.apache.hadoop.fs.s3a.S3A`.

Memulai menggunakan Amazon EMR dengan Amazon S3 di Outposts

Topik berikut menjelaskan cara memulai menggunakan Amazon EMR dengan Amazon S3 di Outposts.

Topik

- Membuat kebijakan izin
- Buat dan konfigurasikan cluster Anda
- Ikhtisar konfigurasi
- Pertimbangan

Membuat kebijakan izin

Sebelum Anda dapat membuat klaster EMR yang menggunakan Amazon S3 di Outposts, Anda harus membuat kebijakan IAM untuk melampirkan ke profil instans Amazon EC2 untuk cluster. Kebijakan harus memiliki izin untuk mengakses jalur akses S3 di Outposts Nama Sumber Daya Amazon (ARN). Untuk informasi selengkapnya tentang membuat kebijakan IAM untuk S3 di Outposts, lihat. Mengatur IAM dengan S3 di Outposts

Kebijakan contoh berikut menunjukkan cara memberikan izin yang diperlukan. Setelah membuat kebijakan, lampirkan kebijakan ke peran profil instance yang Anda gunakan untuk membuat klaster EMR, seperti yang dijelaskan di bagian inithe section called “Buat dan konfigurasikan cluster Anda”. Untuk menggunakan contoh ini, ganti *user input placeholders* dengan informasi Anda sendiri.

```
{
"Version":"2012-10-17",
  "Statement": [
        {
  "Effect": "Allow",
            "Resource": "arn:aws:s3-outposts:us-
west-2:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/access-point-name,
            "Action": [
                "s3-outposts:*"
            ]
        }
    ]

 }
```

Buat dan konfigurasikan cluster Anda

Untuk membuat cluster yang menjalankan Spark dengan S3 di Outposts, selesaikan langkah-langkah berikut di konsol.

Untuk membuat cluster yang berjalan Spark dengan S3 di Outposts

1. Buka konsol Amazon EMR di https://console.aws.amazon.com/elasticmapreduce/.
2. Pada panel navigasi sebelah kiri, pilih Klaster.
3. Pilih Buat klaster.
4. Untuk rilis Amazon EMR, pilih emr-7.0.0atau yang lebih baru.
5. Untuk bundel Aplikasi, pilih Spark interaktif. Kemudian pilih aplikasi lain yang didukung yang ingin Anda sertakan dalam cluster Anda.
6. Untuk mengaktifkan Amazon S3 di Outposts, masukkan pengaturan konfigurasi Anda.

   Contoh pengaturan konfigurasi

   Untuk menggunakan pengaturan konfigurasi sampel berikut, ganti *user input placeholders* dengan informasi Anda sendiri.

```
[
 {
   "Classification": "core-site",
   "Properties": {
     "fs.s3a.bucket.DOC-EXAMPLE-BUCKET.accesspoint.arn": "arn:aws:s3-outposts:us-
west-2:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/access-point-name"
     "fs.s3a.committer.name": "magic",
     "fs.s3a.select.enabled": "false"
    }
  },
  {
    "Classification": "hadoop-env",
    "Configurations": [
      {
        "Classification": "export",
        "Properties": {
          "JAVA_HOME": "/usr/lib/jvm/java-11-amazon-corretto.x86_64"
          }
       }
     ],
     "Properties": {}
   },
   {
     "Classification": "spark-env",
     "Configurations": [
```

```
        {
          "Classification": "export",
          "Properties": {
            "JAVA_HOME": "/usr/lib/jvm/java-11-amazon-corretto.x86_64"
          }
        }
      ],
      "Properties": {}
     },
     {
      "Classification": "spark-defaults",
      "Properties": {
        "spark.executorEnv.JAVA_HOME": "/usr/lib/jvm/java-11-amazon-
corretto.x86_64",
        "spark.sql.sources.fastS3PartitionDiscovery.enabled": "false"
      }
     }
  ]
```

7. Di bagian Networking, pilih virtual private cloud (VPC) dan subnet yang ada di rak Anda. AWS Outposts Untuk informasi selengkapnya tentang Amazon EMR di Outposts, lihat kluster EMR di Panduan Manajemen EMR Amazon. AWS Outposts
8. Di bagian profil instans EC2 untuk Amazon EMR, pilih peran IAM yang memiliki kebijakan izin yang telah dilampirkan sebelumnya.
9. Konfigurasikan setelan cluster Anda yang tersisa, lalu pilih Create cluster.

Ikhtisar konfigurasi

Tabel berikut menjelaskan S3A dan Spark konfigurasi serta nilai yang akan ditentukan untuk parameternya saat Anda menyiapkan cluster yang menggunakan S3 di Outposts dengan Amazon EMR.

Konfigurasi S3A

Parameter	Nilai default	Nilai yang diperlukan untuk S3 di Outposts	Penjelasan
`fs.s3a.aws.credentials.provider`	Jika tidak ditentukan, S3A akan mencari S3 di bucket Region	Titik akses ARN dari ember S3 on Outposts	Amazon S3 di Outposts mendukung titik akses khusus

Parameter	Nilai default	Nilai yang diperlukan untuk S3 di Outposts	Penjelasan
	dengan nama bucket Outposts.		virtual private cloud (VPC) sebagai satu-satunya cara untuk mengakses bucket Outposts.
`fs.s3a.committer.name`	`file`	`magic`	Magic committer adalah satu-satunya committer yang didukung untuk S3 di Outposts.
`fs.s3a.select.enabled`	`TRUE`	`FALSE`	S3 Select tidak didukung di Outposts.
`JAVA_HOME`	`/usr/lib/jvm/java-8`	`/usr/lib/jvm/java-11-amazon-corretto.x86_64`	S3 di Outposts di Java S3A membutuhkan versi 11.

Konfigurasi percikan

Parameter	Nilai default	Nilai yang diperlukan untuk S3 di Outposts	Penjelasan
`spark.sql.sources.fastS3PartitionDiscovery.enabled`	`TRUE`	`FALSE`	S3 di Outposts tidak mendukung partisi cepat.
`spark.executorEnv.JAVA_HOME`	`/usr/lib/jvm/java-8`	`/usr/lib/jvm/java-11-amazon-corretto.x86_64`	S3 di Outposts pada S3A membutuhkan Java versi 11.

Pertimbangan

Pertimbangkan hal berikut saat Anda mengintegrasikan Amazon EMR dengan S3 pada bucket Outposts:

- Amazon S3 di Outposts didukung dengan Amazon EMR versi 7.0.0 dan yang lebih baru.
- Konektor S3A diperlukan untuk menggunakan S3 di Outposts dengan Amazon EMR. Hanya S3A yang memiliki fitur yang diperlukan untuk berinteraksi dengan S3 pada bucket Outposts. Untuk informasi penyiapan konektor S3A, lihat Prasyarat.
- Amazon S3 di Outposts hanya mendukung enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3) dengan Amazon EMR. Untuk informasi selengkapnya, lihat the section called “Enkripsi data”.
- Amazon S3 di Outposts tidak mendukung penulisan dengan S3A. FileOutputCommitter Menulis dengan S3A FileOutputCommitter pada S3 pada bucket Outposts menghasilkan kesalahan berikut: InvalidStorageClass: Kelas penyimpanan yang Anda tentukan tidak valid.
- Amazon S3 di Outposts tidak didukung dengan Amazon EMR Tanpa Server atau Amazon EMR di EKS.
- Log EMR Amazon disimpan di lokasi Amazon S3 Regional yang Anda pilih, dan tidak disimpan secara lokal di bucket S3 on Outposts.

Caching otorisasi dan otentikasi

S3 di Outposts dengan aman menyimpan data otentikasi dan otorisasi secara lokal di rak Outposts. Cache menghapus perjalanan pulang pergi ke induk Wilayah AWS untuk setiap permintaan. Ini menghilangkan variabilitas yang diperkenalkan oleh perjalanan pulang pergi jaringan. Dengan cache otentikasi dan otorisasi di S3 di Outposts, Anda mendapatkan latensi konsisten yang independen dari latensi koneksi antara Outposts dan Outposts. Wilayah AWS

Saat Anda membuat permintaan API S3 on Outposts, data autentikasi dan otorisasi di-cache dengan aman. Data yang di-cache kemudian digunakan untuk mengautentikasi permintaan API objek S3 berikutnya. S3 di Outposts hanya menyimpan data otentikasi dan otorisasi cache saat permintaan ditandatangani menggunakan Signature Version 4A (Sigv4a). Cache disimpan secara lokal di Outposts dalam layanan S3 on Outposts. Ini menyegarkan secara asinkron saat Anda membuat permintaan API S3. Cache dienkripsi, dan tidak ada kunci kriptografi plaintext yang disimpan di Outposts.

Cache berlaku hingga 10 menit ketika Outpost terhubung ke file. Wilayah AWS Ini disegarkan secara asinkron saat Anda membuat permintaan S3 pada Outposts API, untuk memastikan bahwa kebijakan terbaru digunakan. Jika Outpost terputus dari Wilayah AWS, cache akan berlaku hingga 12 jam.

Mengkonfigurasi cache otorisasi dan otentikasi

S3 di Outposts secara otomatis menyimpan data otentikasi dan otorisasi untuk permintaan yang ditandatangani dengan algoritma Sigv4a. Untuk informasi selengkapnya, lihat Menandatangani permintaan AWS API di Panduan AWS Identity and Access Management Pengguna. Algoritma Sigv4a tersedia dalam versi terbaru SDK. AWS Anda dapat memperolehnya melalui ketergantungan pada pustaka AWS Common Runtime (CRT).

Anda perlu menggunakan AWS SDK versi terbaru dan menginstal CRT versi terbaru. Misalnya, Anda dapat menjalankan `pip install awscrt` untuk mendapatkan CRT versi terbaru dengan Boto3.

S3 di Outposts tidak menyimpan data otentikasi dan otorisasi cache untuk permintaan yang ditandatangani dengan algoritma SiGv4.

Memvalidasi penandatanganan Sigv4a

Anda dapat menggunakan AWS CloudTrail untuk memvalidasi bahwa permintaan telah ditandatangani dengan Sigv4a. Untuk informasi lebih lanjut tentang pengaturan CloudTrail untuk S3 di Outposts, lihat. Memantau S3 di AWS CloudTrail Outposts dengan log

Setelah Anda mengonfigurasi CloudTrail, Anda dapat memverifikasi bagaimana permintaan ditandatangani di `SignatureVersion` bidang CloudTrail log. Permintaan yang ditandatangani dengan Sigv4a akan disetel ke`SignatureVersion`. `AWS 4-ECDSA-P256-SHA256` Permintaan yang ditandatangani dengan SiGv4 akan `SignatureVersion` disetel ke. `AWS 4-HMAC-SHA256`

Keamanan di S3 di Outposts

Keamanan cloud di AWS merupakan prioritas tertinggi. Sebagai pelanggan AWS, Anda mendapatkan manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara AWS dan Anda. Model tanggung jawab bersama menjelaskan hal ini sebagai keamanan dari cloud dan keamanan dalam cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang berjalan Layanan AWS di dalamnyaAWS Cloud. AWSjuga memberi Anda layanan yang dapat Anda gunakan dengan

aman. Auditor pihak ketiga melakukan pengujian dan verifikasi secara berkala terhadap efektivitas keamanan kami sebagai bagian dari Program Kepatuhan AWS.

- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh Layanan AWS yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, termasuk sensitivitas data Anda, persyaratan perusahaan Anda, serta undang-undang dan peraturan yang berlaku.

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan S3 di Outposts. Topik berikut menunjukkan cara mengonfigurasi S3 di Outposts untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga belajar cara menggunakan Layanan AWS yang lain yang membantu Anda memantau dan mengamankan sumber daya S3 Anda di Outposts.

Topik

- Enkripsi data di S3 di Outposts
- AWS PrivateLink untuk S3 di Outposts
- AWSKunci kebijakan autentikasi Versi 4 (SiGv4)
- AWSkebijakan terkelola untuk Amazon S3 di Outposts
- Menggunakan peran terkait layanan untuk Amazon S3 di Outposts

Enkripsi data di S3 di Outposts

Secara default, semua data yang disimpan di Amazon S3 di Outposts dienkripsi dengan menggunakan enkripsi sisi server dengan kunci enkripsi terkelola Amazon S3 (SSE-S3). Untuk informasi selengkapnya, lihat Menggunakan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3).

Anda juga dapat menggunakan enkripsi sisi server dengan kunci enkripsi yang disediakan pelanggan (SSE-C). Untuk menggunakan SSE-C, tentukan kunci enkripsi sebagai bagian dari permintaan API objek Anda. Enkripsi sisi server hanya mengenkripsi data objek, bukan metadata objek. Untuk informasi selengkapnya, lihat Menggunakan enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C).

> **Note**
>
> S3 di Outposts tidak mendukung enkripsi sisi serverAWS Key Management Service(AWS KMS) kunci (SSE-KMS).

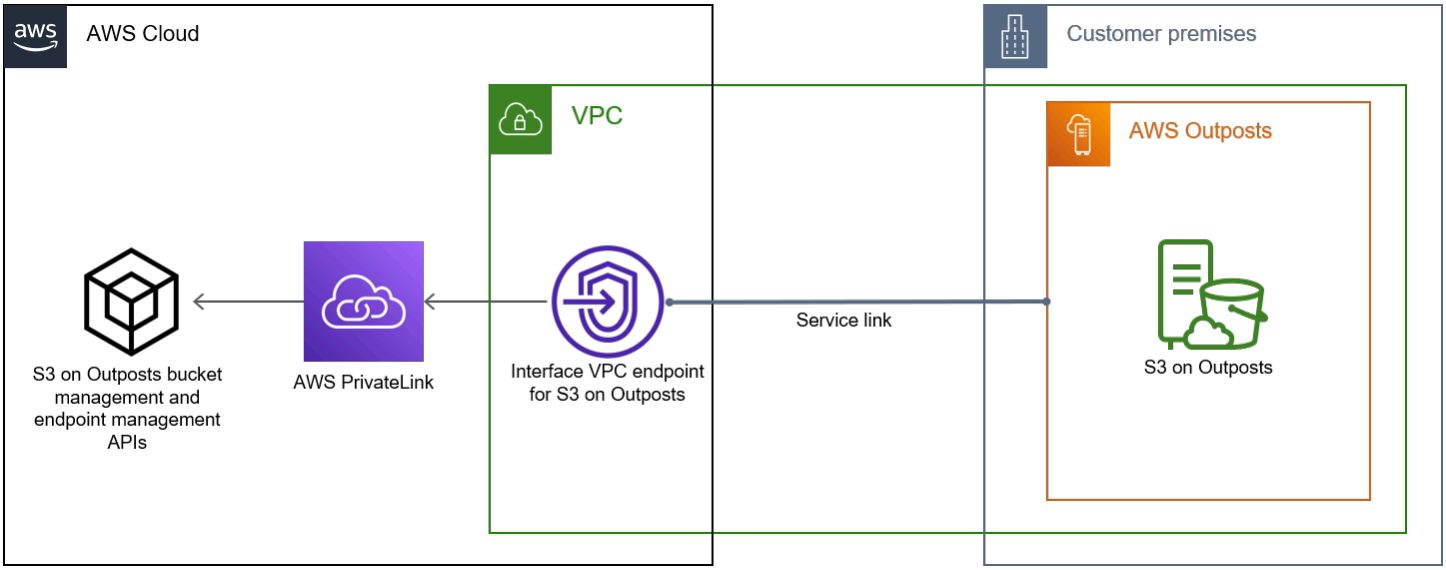
AWS PrivateLink untuk S3 di Outposts

S3 on Outposts AWS PrivateLink mendukung, yang menyediakan akses manajemen langsung ke S3 Anda pada penyimpanan Outposts melalui titik akhir pribadi dalam jaringan pribadi virtual Anda. Ini memungkinkan Anda untuk menyederhanakan arsitektur jaringan internal Anda dan melakukan operasi manajemen pada penyimpanan objek Outposts Anda dengan menggunakan alamat IP pribadi di Virtual Private Cloud (VPC) Anda. Menggunakan AWS PrivateLink menghilangkan kebutuhan untuk menggunakan alamat IP publik atau server proxy.

Dengan AWS PrivateLink Amazon S3 di Outposts, Anda dapat menyediakan endpoint VPC antarmuka di virtual private cloud (VPC) Anda untuk mengakses S3 Anda di Outposts bucket management dan endpoint management API. Titik akhir VPC antarmuka dapat diakses langsung dari aplikasi yang digunakan di VPC Anda atau di tempat melalui jaringan privat virtual (VPN) Anda atau. AWS Direct Connect Anda dapat mengakses bucket dan endpoint management API melalui AWS PrivateLink. AWS PrivateLink tidak mendukung operasi API transfer data, seperti GET, PUT, dan API serupa. Operasi ini sudah ditransfer secara pribadi melalui titik akhir S3 pada Outposts dan konfigurasi titik akses. Untuk informasi selengkapnya, lihat Jaringan untuk S3 di Outposts.

Titik akhir antarmuka diwakili oleh satu atau lebih antarmuka jaringan elastis (ENI) yang ditugaskan alamat IP privat dari subnet di VPC Anda. Permintaan yang dibuat ke titik akhir antarmuka untuk S3 di Outposts secara otomatis dirutekan ke S3 pada bucket Outposts dan API manajemen titik akhir di jaringan. AWS Anda juga dapat mengakses titik akhir antarmuka di VPC Anda dari aplikasi lokal AWS Direct Connect melalui AWS Virtual Private Network atau ().AWS VPN Untuk informasi selengkapnya tentang cara menghubungkan VPC dengan jaringan on-premise Anda, lihat AWS Direct Connect Panduan Pengguna dan AWS Site-to-Site VPN Panduan Pengguna.

Permintaan rute titik akhir antarmuka untuk S3 pada bucket Outposts dan API manajemen endpoint melalui AWS jaringan dan melalui AWS PrivateLink, seperti yang diilustrasikan dalam diagram berikut.



Untuk informasi umum tentang titik akhir antarmuka, lihat Antarmuka titik akhir VPC (AWS PrivateLink) dalam Panduan AWS PrivateLink .

Topik

- Pembatasan dan batasan
- Mengakses S3 di Outposts
- Memperbarui konfigurasi DNS on-premise
- Membuat titik akhir VPC untuk S3 di Outposts
- Membuat kebijakan bucket dan kebijakan titik akhir VPC untuk S3 di Outposts

Pembatasan dan batasan

Saat Anda mengakses S3 di bucket Outposts dan API manajemen endpoint AWS PrivateLink, pembatasan VPC berlaku. Untuk informasi selengkapnya, lihat Properti titik akhir antarmuka dan batas dan AWS PrivateLink kuota di AWS PrivateLink Panduan.

Selain itu, AWS PrivateLink tidak mendukung yang berikut:

- Titik Akhir Standar Proses Informasi Federal (FIPS)
- S3 pada API transfer data Outposts, misalnya, GET, PUT, dan operasi API objek serupa.
- DNS privat

Mengakses S3 di Outposts

Untuk mengakses S3 pada bucket Outposts dan API manajemen endpoint AWS PrivateLink menggunakan, Anda harus memperbarui aplikasi Anda untuk menggunakan nama DNS khusus titik akhir. Saat Anda membuat titik akhir antarmuka, AWS PrivateLink buat dua jenis S3 khusus titik akhir pada nama Outposts: Regional dan zonal.

- Nama DNS regional — termasuk ID titik akhir VPC unik, pengenal layanan, `vpce.amazonaws.com` dan, Wilayah AWS misalnya,. *`vpce-1a2b3c4d-5e6f`*`.s3-outposts.us-east-1.vpce.amazonaws.com`
- Nama DNS zona — termasuk ID titik akhir VPC unik, Availability Zone, pengenal layanan, dan, misalnya Wilayah AWS,. `vpce.amazonaws.com` *`vpce-1a2b3c4d-5e6f`*`-`**`us-east-1a.`**`s3-outposts.us-east-1.vpce.amazonaws.com` Anda dapat menggunakan opsi ini jika arsitektur Anda mengisolasi Zona Ketersediaan. Misalnya, Anda bisa menggunakan nama DNS zonal untuk penahanan kesalahan atau untuk mengurangi biaya transfer data Regional.

> **Important**
>
> S3 pada titik akhir antarmuka Outposts diselesaikan dari domain DNS publik. S3 di Outposts tidak mendukung DNS pribadi. Gunakan `--endpoint-url` parameter untuk semua API manajemen bucket dan titik akhir.

AWS CLI contoh

Gunakan `--endpoint-url` parameter `--region` dan untuk mengakses manajemen bucket dan API manajemen titik akhir melalui S3 pada titik akhir antarmuka Outposts.

Example : Gunakan URL titik akhir untuk daftar bucket dengan API kontrol S3

Dalam contoh berikut, ganti Wilayah *`us-east-1`*, URL titik akhir VPC*`vpce-1a2b3c4d-5e6f.s3.us-east-1.vpce.amazonaws.com`*, dan ID akun *`111122223333`* dengan informasi yang sesuai.

```
aws s3control list-regional-buckets --region us-east-1 --endpoint-url
 https://vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com --account-
id 111122223333
```

AWS Contoh SDK

Perbarui SDK Anda ke versi terbaru, dan konfigurasikan klien Anda untuk menggunakan URL titik akhir untuk mengakses API kontrol S3 untuk titik akhir antarmuka Outposts. Untuk informasi selengkapnya, lihat contoh AWS SDK untuk AWS PrivateLink.

SDK for Python (Boto3)

Example : Gunakan URL titik akhir untuk mengakses API kontrol S3

Dalam contoh berikut, ganti URL titik akhir Wilayah *us-east-1* dan VPC *vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com* dengan informasi yang sesuai.

```
control_client = session.client(
service_name='s3control',
region_name='us-east-1',
endpoint_url='https://vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com'
)
```

Untuk informasi selengkapnya, lihat AWS PrivateLink Amazon S3 di panduan pengembang Boto3.

SDK for Java 2.x

Example : Gunakan URL titik akhir untuk mengakses API kontrol S3

Dalam contoh berikut, ganti URL titik akhir VPC *vpce-1a2b3c4d-5e6f.s3-outposts.us-east-1.vpce.amazonaws.com* dan Wilayah *Region.US_EAST_1* dengan informasi yang sesuai.

```
// control client
Region region = Region.US_EAST_1;
s3ControlClient = S3ControlClient.builder().region(region)

 .endpointOverride(URI.create("https://vpce-1a2b3c4d-5e6f.s3-outposts.us-
east-1.vpce.amazonaws.com"))
                                .build()
```

Untuk informasi selengkapnya, lihat `S3ControlClient` dalam Referensi API AWS SDK for Java .

Memperbarui konfigurasi DNS on-premise

Saat menggunakan nama DNS titik akhir untuk mengakses titik akhir antarmuka untuk S3 di Outposts bucket management dan titik akhir API, Anda tidak perlu memperbarui penyelesai DNS on-premise Anda. Anda dapat menyelesaikan nama DNS titik akhir khusus dengan alamat IP privat titik akhir antarmuka dari domain S3 publik di Outposts.

Membuat titik akhir VPC untuk S3 di Outposts

Untuk membuat titik akhir antarmuka VPC untuk S3 di Outposts, lihat Membuat titik akhir VPC di Panduan.AWS PrivateLink

Membuat kebijakan bucket dan kebijakan titik akhir VPC untuk S3 di Outposts

Anda dapat melampirkan kebijakan titik akhir ke titik akhir VPC yang mengontrol akses ke S3 di Outposts. Anda juga dapat menggunakan kebijakan `aws:sourceVpce` bucket S3 di Outposts untuk membatasi akses ke bucket tertentu dari titik akhir VPC tertentu. Dengan kebijakan titik akhir VPC, Anda dapat mengontrol akses ke S3 pada API manajemen bucket Outposts dan API manajemen titik akhir. Dengan kebijakan bucket, Anda dapat mengontrol akses ke API manajemen bucket S3 on Outposts. Namun, Anda tidak dapat mengelola akses ke tindakan objek untuk S3 di `aws:sourceVpce` Outposts menggunakan.

Kebijakan akses untuk S3 di Outposts menentukan informasi berikut:

- Prinsip AWS Identity and Access Management (IAM) yang tindakannya diizinkan atau ditolak.
- Tindakan kontrol S3 yang diizinkan atau ditolak.
- Sumber daya S3 di Outposts di mana tindakan diizinkan atau ditolak.

Contoh berikut menunjukkan kebijakan yang membatasi akses ke bucket atau titik akhir. Untuk informasi selengkapnya tentang konektivitas VPC, lihat opsi konektivitas jaringan-ke-VPC di whitepaper Opsi Konektivitas Amazon AWS Virtual Private Cloud.

> **Important**
>
> - Saat menerapkan kebijakan contoh untuk titik akhir VPC yang dijelaskan di bagian ini, Anda dapat memblokir akses Anda ke bucket tanpa bermaksud melakukannya. Izin bucket yang membatasi akses bucket ke koneksi yang berasal dari titik akhir VPC Anda dapat memblokir semua koneksi ke bucket tersebut. Untuk informasi tentang cara mengatasi

masalah ini, lihat Kebijakan bucket saya memiliki ID titik akhir VPC atau VPC yang salah. Bagaimana saya dapat memperbaiki kebijakan tersebut sehingga saya dapat mengakses bucket? dalam Pusat Pengetahuan AWS Support .

- Sebelum menggunakan kebijakan bucket contoh berikut ini, ganti ID titik akhir VPC dengan nilai yang sesuai untuk kasus penggunaan Anda. Jika tidak, Anda tidak akan dapat mengakses bucket Anda.
- Jika kebijakan Anda hanya mengizinkan akses ke bucket S3 di Outposts dari titik akhir VPC tertentu, kebijakan tersebut nonaktifkan akses konsol untuk ember tersebut karena permintaan konsol tidak berasal dari titik akhir VPC tertentu.

Topik

- Contoh: Membatasi Akses ke bucket khusus dari titik akhir VPC
- Contoh: Menolak Akses dari titik akhir VPC tertentu di kebijakan bucket S3 di Outposts

Contoh: Membatasi Akses ke bucket khusus dari titik akhir VPC

Anda dapat membuat kebijakan titik akhir yang membatasi akses ke bucket S3 tertentu di Outposts saja. Kebijakan berikut membatasi akses untuk GetBucketPolicy tindakan hanya ke. *example-outpost-bucket* Untuk menggunakan kebijakan ini, ganti nilai contoh dengan kebijakan Anda sendiri.

```
{
  "Version": "2012-10-17",
  "Id": "Policy1415115909151",
  "Statement": [
    { "Sid": "Access-to-specific-bucket-only",
      "Principal": {"AWS":"111122223333"},
      "Action": "s3-outposts:GetBucketPolicy",
      "Effect": "Allow",
      "Resource": "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outpost-
bucket"
    }
  ]
}
```

Contoh: Menolak Akses dari titik akhir VPC tertentu di kebijakan bucket S3 di Outposts

Kebijakan bucket S3 on Outposts berikut menolak akses GetBucketPolicy ke bucket melalui endpoint *example-outpost-bucket* VPC. *vpce-1a2b3c4d*

`aws:sourceVpce`Syarat menentukan titik akhir dan tidak memerlukan Amazon Resource Name (ARN) untuk sumber daya titik akhir VPC, tetapi hanya ID titik akhir. Untuk menggunakan kebijakan ini, ganti nilai contoh dengan kebijakan Anda sendiri.

```
{
    "Version": "2012-10-17",
    "Id": "Policy1415115909152",
    "Statement": [
        {
            "Sid": "Deny-access-to-specific-VPCE",
            "Principal": {"AWS":"111122223333"},
            "Action": "s3-outposts:GetBucketPolicy",
            "Effect": "Deny",
            "Resource": "arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outpost-
bucket",
            "Condition": {
                "StringEquals": {"aws:sourceVpce": "vpce-1a2b3c4d"}
            }
        }
    ]
}
```

AWSKunci kebijakan autentikasi Versi 4 (SiGv4)

Tabel berikut menunjukkan kunci kondisi yang terkait denganAWSAutentikasi Signature Version 4 (Signature Version 4) yang dapat Anda gunakan dengan Amazon S3 di Outposts. Dalam kebijakan kelompok, Anda dapat menambahkan ketentuan ini untuk menerapkan perilaku tertentu saat permintaan diautentikasi dengan menggunakan Signature Version 4. Untuk kebijakan-kebijakan contoh, lihat Contoh kebijakan bucket yang menggunakan kunci kondisi terkait Signature Version 4. Untuk informasi lebih lanjut tentang mengautentikasi permintaan menggunakan Signature Version 4, lihatMengautentikasi permintaan (AWSTanda Tangan Versi 4)di dalamReferensi API Amazon Simple Storage Service

Kunci yang berlaku untuk**`s3-outposts:*`**tindakan atau tindakan S3 pada Outposts

Kunci yang berlaku	Deskripsi

`s3-outposts:authType`	S3 di Outposts mendukung berbagai metode autentikasi. Untuk membatasi permintaan masuk agar menggunakan metode otentikasi tertentu, Anda dapat menggunakan kunci kondisi opsional ini. Misalnya, Anda dapat menggunakan kunci syarat ini untuk mengizinkan hanya HTTP`Authorization` header yang akan digunakan dalam otentikasi permintaan.
	
	Nilai yang valid:
	
	`REST-HEADER`
	
	`REST-QUERY-STRING`

`s3-outposts:signatureAge`	Lamanya waktu, dalam milidetik, tanda tangan valid dalam permintaan yang diautentikasi. Kondisi ini hanya berfungsi untuk URL yang telah ditandatangani sebelumnya. Di Signature Version 4, kunci penandatangan berlaku hingga tujuh hari. Oleh karena itu, tanda tangan juga berlaku hingga tujuh hari. Untuk informasi selengkapnya, lihat[Pengantar permintaan penandatangan]di dalamReferensi API Amazon Simple Storage Service. Anda dapat menggunakan kondisi ini untuk lebih membatasi usia tanda tangan. Nilai contoh: `600000`
`s3-outposts:x-amz-content-sha256`	Anda dapat menggunakan kunci kondisi ini untuk melarang konten yang tidak ditandatangani di bucket Anda. Saat Anda menggunakan Signature Version 4, untuk permintaan yang menggunakan`Authorization` header, Anda menambahkan`x-amz-content-sha256` header dalam perhitungan tanda tangan dan kemudian mengatur nilainya ke payload hash.

Kunci yang berlaku	Deskripsi
	Anda dapat menggunakan kunci kondisi ini dalam kebijakan bucket Anda untuk menolak upload apa pun yang payload tidak ditandatangani. Misalnya: • Tolak unggahan yang menggunakan`Authorization` header untuk mengotentikasi permintaan tetapi tidak menandatangani payload. Untuk informasi selengkapnya, lihatMentransfer muatan dalam satu bagiandi dalamReferensi API Amazon Simple Storage Service. • Tolak upload yang menggunakan URL yang telah ditandata ngani sebelumnya. URL yang ditandatangani sebelumnya selalu memilikiUNSIGNED_PAYLOAD . Untuk informasi selengkap nya, lihatMengautentikasi permintaandanMetode Autentikasidi dalamReferensi API Amazon Simple Storage Service. Nilai yang valid: `UNSIGNED-PAYLOAD`

Contoh kebijakan bucket yang menggunakan kunci kondisi terkait Signature Version 4

Untuk menggunakan contoh berikut, ganti*user input placeholders*dengan informasi Anda sendiri.

Example : **`s3-outposts:signatureAge`**

Kebijakan bucket berikut menolak permintaan URL S3 pada Outposts yang ditandatangani sebelumnya pada objek di`example-outpost-bucket`jika tanda tangan berusia lebih dari 10 menit.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Deny a presigned URL request if the signature is more than 10
 minutes old",
            "Effect": "Deny",
            "Principal": {"AWS":"444455556666"},
            "Action": "s3-outposts:*",
```

```
            "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/object/
*",
            "Condition": {
                "NumericGreaterThan": {"s3-outposts:signatureAge": 600000},
                "StringEquals": {"s3-outposts:authType": "REST-QUERY-STRING"}
            }
        }
    ]
}
```

Example : **`s3-outposts:authType`**

Kebijakan bucket berikut hanya mengizinkan permintaan yang menggunakan`Authorization`header untuk otentikasi permintaan. Setiap permintaan URL yang telah ditandatangani sebelumnya akan ditolak karena URL yang telah ditandatangani sebelumnya menggunakan parameter kueri untuk memberikan informasi permintaan dan otentikasi. Untuk informasi selengkapnya, lihatMetode Autentikasidi dalamReferensi API Amazon Simple Storage Service.

```
{
   "Version": "2012-10-17",
   "Statement": [
         {
               "Sid": "Allow only requests that use the Authorization header for
 request authentication. Deny presigned URL requests.",
               "Effect": "Deny",
               "Principal": {"AWS":"111122223333"},
               "Action": "s3-outposts:*",
               "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/object/
*",
               "Condition": {
                     "StringNotEquals": {
                           "s3-outposts:authType": "REST-HEADER"
                     }
               }
         }
   ]
}
```

Example : **`s3-outposts:x-amz-content-sha256`**

Kebijakan bucket berikut menolak upload apa pun dengan payload yang tidak ditandatangani, seperti upload yang menggunakan URL yang ditandatangani sebelumnya. Untuk informasi selengkapnya, lihatMengautentikasi permintaandanMetode Autentikasidi dalamReferensi API Amazon Simple Storage Service.

```
{
   "Version": "2012-10-17",
   "Statement": [
         {
               "Sid": "Deny uploads with unsigned payloads.",
               "Effect": "Deny",
               "Principal": {"AWS":"111122223333"},
               "Action": "s3-outposts:*",
               "Resource": "arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/bucket/example-outpost-bucket/object/
*",
               "Condition": {
                     "StringEquals": {
                           "s3-outposts:x-amz-content-sha256": "UNSIGNED-PAYLOAD"
                     }
               }
         }
   ]
}
```

AWSkebijakan terkelola untuk Amazon S3 di Outposts

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola olehAWS. AWSKebijakan terkelola dirancang untuk memberikan izin bagi banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan semua pelanggan. AWS Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pemutakhiran akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut.

AWSkemungkinan besar akan memperbarui kebijakan AWS terkelola saat baru Layanan AWS diluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat Kebijakan terkelola AWS dalam Panduan Pengguna IAM.

AWSkebijakan terkelola: AWSS3OnOutpostsServiceRolePolicy

Membantu mengelola sumber daya jaringan untuk Anda sebagai bagian dari peran terkait layanan. `AWSServiceRoleForS3OnOutposts`

Untuk melihat izin kebijakan ini, lihat AWSS3OnOutpostsServiceRolePolicy.

Pembaruan AWS S3 tentang Outposts ke kebijakan terkelola

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk S3 di Outposts sejak layanan ini mulai melacak perubahan ini.

Perubahan	Deskripsi	Tanggal
S3 di Outposts ditambahkan `AWSS3OnOutpostsServiceRolePolicy`	S3 di Outposts `AWSS3OnOutpostsServiceRolePolicy` ditambahkan sebagai bagian dari `AWSServiceRoleForS3OnOutposts` peran terkait layanan, yang membantu mengelola sumber daya jaringan untuk Anda.	Oktober 3, 2023
S3 di Outposts mulai melacak perubahan	S3 di Outposts mulai melacak perubahan untuk AWS kebijakan yang dikelola.	Oktober 3, 2023

Menggunakan peran terkait layanan untuk Amazon S3 di Outposts

Amazon S3 di Outposts menggunakan peran terkait layanan AWS Identity and Access Management (IAM). Peran terkait layanan adalah jenis peran IAM unik yang ditautkan langsung ke S3 di Outposts. Peran terkait layanan telah ditentukan sebelumnya oleh S3 di Outposts dan menyertakan semua izin yang diperlukan layanan untuk memanggil layanan lain atas nama Anda. AWS

Peran terkait layanan membuat pengaturan S3 di Outposts lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. S3 di Outposts mendefinisikan izin dari peran terkait layanannya, dan kecuali ditentukan lain, hanya S3 di Outposts yang dapat mengambil perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran tertaut layanan hanya setelah menghapus sumber daya terkait terlebih dahulu. Ini melindungi sumber daya S3 Anda di Outposts karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran terkait layanan, lihat AWSLayanan yang Bekerja dengan IAM dan cari layanan yang memiliki Ya di kolom Peran terkait layanan. Pilih Ya dengan tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Izin peran terkait layanan untuk S3 di Outposts

S3 di Outposts menggunakan peran terkait layanan AWSServiceRoleForS3OnOutpostsbernama untuk membantu mengelola sumber daya jaringan untuk Anda.

`AWSServiceRoleForS3OnOutposts` peran terkait layanan memercayakan layanan berikut untuk menjalankan peran tersebut:

- `s3-outposts.amazonaws.com`

Kebijakan izin peran bernama `AWSS3OnOutpostsServiceRolePolicy` memungkinkan S3 di Outposts untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeVpcs",
                "ec2:DescribeCoipPools",
                "ec2:GetCoipPoolUsage",
                "ec2:DescribeAddresses",
                "ec2:DescribeLocalGatewayRouteTableVpcAssociations"
            ],
```

```
            "Resource": "*",
            "Sid": "DescribeVpcResources"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:security-group/*"
            ],
            "Sid": "CreateNetworkInterface"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "S3 On Outposts"
                }
            },
            "Sid": "CreateTagsForCreateNetworkInterface"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AllocateAddress"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:ipv4pool-ec2/*"
            ],
            "Sid": "AllocateIpAddress"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AllocateAddress"
            ],
```

```
            "Resource": [
                "arn:aws:ec2:*:*:elastic-ip/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CreatedBy": "S3 On Outposts"
                }
            },
            "Sid": "CreateTagsForAllocateIpAddress"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:CreateNetworkInterfacePermission",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteNetworkInterfacePermission",
                "ec2:DisassociateAddress",
                "ec2:ReleaseAddress",
                "ec2:AssociateAddress"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/CreatedBy": "S3 On Outposts"
                }
            },
            "Sid": "ReleaseVpcResources"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": [
                        "CreateNetworkInterface",
                        "AllocateAddress"
                    ],
                    "aws:RequestTag/CreatedBy": [
                        "S3 On Outposts"
                    ]
```

```
                }
            },
            "Sid": "CreateTags"
        }
    ]
}
```

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti peran) membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, lihat Izin peran tertaut layanan dalam Panduan Pengguna IAM.

Membuat peran terkait layanan untuk S3 di Outposts

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda membuat titik akhir S3 di Outposts di, AWS Management Console AWS CLI the, atau AWS API, S3 di Outposts membuat peran terkait layanan untuk Anda.

Jika Anda menghapus peran tertaut layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Saat Anda membuat S3 pada titik akhir Outposts, S3 di Outposts membuat peran terkait layanan untuk Anda lagi.

Anda juga dapat menggunakan konsol IAM untuk membuat peran terkait layanan dengan kasus penggunaan S3 on Outposts. Di AWS CLI atau API AWS, buat peran yang terhubung dengan layanan dengan nama layanan `s3-outposts.amazonaws.com`. Untuk informasi lebih lanjut, lihat Membuat peran terkait layanan dalam Panduan Pengguna IAM. Jika Anda menghapus peran tertaut layanan ini, Anda dapat mengulang proses yang sama untuk membuat peran tersebut lagi.

Mengedit peran terkait layanan untuk S3 di Outposts

S3 di Outposts tidak memungkinkan Anda mengedit peran terkait layanan`AWSServiceRoleForS3OnOutposts`. Ini termasuk nama peran karena berbagai entitas mungkin merujuknya. Namun, Anda dapat mengedit penjelasan peran menggunakan IAM. Untuk informasi selengkapnya, lihat Mengedit peran yang terkait dengan layanan dalam Panduan Pengguna IAM.

Menghapus peran terkait layanan untuk S3 di Outposts

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran terkait layanan, kami merekomendasikan Anda menghapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas yang tidak digunakan yang tidak dipantau atau dipelihara secara aktif. Tetapi, Anda harus

membersihkan sumber daya peran yang terhubung dengan layanan sebelum menghapusnya secara manual.

> **Note**
>
> Jika layanan S3 on Outposts menggunakan peran saat Anda mencoba menghapus sumber daya, maka penghapusan mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk menghapus S3 pada sumber daya Outposts yang digunakan oleh peran AWSServiceRoleForS3OnOutposts

1. Hapus titik akhir S3 di Outposts di Akun AWS seluruh Anda. Wilayah AWS
2. Hapus peran terkait layanan menggunakan IAM.

   Gunakan konsol IAM, AWS CLI, atau AWS API untuk menghapus peran terkait layanan `AWSServiceRoleForS3OnOutposts` Untuk informasi selengkapnya, lihat Menghapus peran tertaut layanan dalam Panduan Pengguna IAM.

Wilayah yang Didukung untuk S3 pada peran terkait layanan Outposts

S3 on Outposts mendukung penggunaan peran terkait layanan di semua tempat layanan Wilayah AWS tersedia. Untuk informasi selengkapnya, lihat S3 di Wilayah Outposts dan titik akhir.

Mengelola penyimpanan S3 di Outposts

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di Outposts AWS Anda dan dengan mudah menyimpan dan mengambil objek on-premise untuk aplikasi yang memerlukan akses data lokal, pengolahan data lokal, dan residensi data. S3 di Outposts menyediakan kelas penyimpanan baru, S3OUTPOSTS APIs, dan dirancang untuk menyimpan data secara Amazon S3 APIs, dan dirancang untuk menyimpan data secara tahan lama dan redundan di beberapa perangkat dan server di AndaAWS Outposts. Anda berkomunikasi dengan bucket Outposts menggunakan titik akses dan koneksi titik akhir melalui cloud pribadi virtual (VPC). Anda dapat menggunakan API dan fitur yang sama di bucket Outposts seperti yang Anda lakukan di bucket Amazon S3, termasuk kebijakan akses, enkripsi, enkripsi, enkripsi, enkripsi, enkripsi, enkripsi, dan pemberian tag. Anda dapat menggunakan S3 di Outposts melaluiAWS Management Console,AWS Command Line

Interface (AWS CLI),AWS SDK, atau REST API. Untuk informasi selengkapnya, lihat Apa itu Amazon S3 di Outposts?

Untuk informasi lebih lanjut tentang mengelola dan berbagi kapasitas penyimpanan Amazon S3 di Outposts, lihat topik berikut.

Topik

- Mengelola versioning
- Membuat siklus aktifnya untuk bucket Amazon S3 di Outposts
- Replikasi objek untuk S3 di Outposts
- Berbagi S3 di Outposts dengan menggunakan AWS RAM
- LainnyaLayanan AWSS3 di Outposts

Mengelola versioning

Saat diaktifkan, versioning ing menyimpan beberapa salinan objek yang berbeda dalam bucket yang sama. Anda dapat menggunakan versioning untuk menyimpan, mengambil, dan memulihkan setiap versi dari setiap objek yang disimpan dalam bucket Outposts S3 Versioning membantu Anda memulihkan dari tindakan pengguna yang tidak diinginkan dan kegagalan aplikasi.

Bucket Amazon S3 di Outposts memiliki tiga status versioning:

- Tidak Berversi - Jika Anda belum pernah mengaktifkan atau menangguhkan Versi S3 di bucket Anda, versi tersebut tidak diversi dan tidak mengembalikan status Versi S3. Untuk informasi selengkapnya tentang S3 Versioning, lihat Menggunakan Penentuan Versi dalam bucket S3.
- Diaktifkan - Memungkinkan Versi S3 untuk objek di bucket. Semua objek yang ditambahkan ke bucket menerima ID versi unik. Objek yang sudah ada di bucket pada saat Anda mengaktifkan versi memiliki ID versi`null`. Jika Anda memodifikasi objek ini (atau lainnya) dengan operasi lain PutObject, seperti, objek baru mendapatkan ID versi unik.
- Ditangguhkan - Menangguhkan Versi S3 untuk objek di bucket. Semua objek yang ditambahkan ke bucket setelah versi ditangguhkan menerima ID versi`null`. Untuk informasi selengkapnya, lihat Menambahkan objek ke bucket dengan Penentuan Versi ditangguhkan.

Setelah Anda mengaktifkan versioning untuk bucket S3 di Outposts, itu tidak akan pernah dapat kembali ke status tidak berversi. Namun, Anda dapat menangguhkan versioning. Untuk informasi selengkapnya tentang S3 Versioning, lihat Menggunakan Penentuan Versi dalam bucket S3.

Untuk setiap objek dalam bucket, Anda memiliki versi saat ini dan nol atau lebih versi non-terkini. Untuk mengurangi biaya penyimpanan, Anda dapat mengonfigurasi aturan Siklus Hidup bucket S3 agar kedaluwarsa versi noncurrent setelah jangka waktu tertentu. Untuk informasi selengkapnya, lihat Membuat siklus aktifnya untuk bucket Amazon S3 di Outposts.

Contoh berikut menunjukkan cara mengaktifkan atau menangguhkan versi untuk bucket S3 on Outposts yang ada dengan menggunakanAWS Management Console danAWS Command Line Interface (AWS CLI). Untuk membuat bucket dengan versioningMembuat bucket S3 di Outposts ing

> **Note**
>
> Akun AWS yang menciptakan bucket memilikinya dan merupakan satu-satunya yang dapat melakukan tindakan untuk itu. Bucket memiliki properti konfigurasi, seperti Outpost, tag, enkripsi default, dan pengaturan titik akses. Setelan titik akses termasuk cloud pribadi virtual (VPC), kebijakan titik akses untuk mengakses objek dalam bucket, dan metadata lainnya. Untuk informasi selengkapnya, lihat S3 di spesifikasi Outposts.

Menggunakan konsol S3

Untuk mengedit pengaturan Versi S3 untuk bucket Anda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih bucket Outposts.
3. Pilih bucket Outposts yang ingin Anda aktifkan versioning ing
4. Pilih tab Properti.
5. Di Bawah Versi Bucket, pilih Edit.
6. Edit setelan versioning untuk bucket dengan memilih salah satu opsi berikut:
   - Untuk menangguhkan Versi S3 dan menghentikan pembuatan versi objek baru, pilih Suspend.
   - Untuk mengaktifkan S3 Versioning dan menyimpan beberapa salinan berbeda dari setiap objek, pilih Enable.
7. Pilih Save changes (Simpan perubahan).

Menggunakan perintah AWS CLI

Untuk mengaktifkan atau menangguhkan Versi S3 untuk bucket Anda dengan menggunakanAWS CLI, gunakan`put-bucket-versioning` perintah, seperti yang ditunjukkan dalam contoh berikut. Untuk menggunakan contoh ini, ganti masing-masing*`user input placeholder`* dengan informasi Anda sendiri.

Untuk informasi selengkapnya, lihat put-bucket-versioning dalam Referensi AWS CLI.

Example : Untuk mengaktifkan versioning

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --versioning-configuration Status=Enabled
```

Example : Untuk menangguhkan Versi S3

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/op-01ac5d28a6a232904/bucket/example-outposts-
bucket --versioning-configuration Status=Suspended
```

Membuat siklus aktifnya untuk bucket Amazon S3 di Outposts

Anda dapat menggunakan aktifnya untuk mengoptimalkan kapasitas penyimpanan untuk Amazon S3 pada Outposts. Anda dapat membuat aturan siklus hidup untuk kedaluwarsa objek seiring bertambahnya usia atau diganti dengan versi yang lebih baru. Anda dapat membuat, mengaktifkan, menonaktifkan, atau menghapus aturan siklus hidup.

Untuk informasi lebih lanjut tentang S3 Lifecycle, lihatMengelola siklus hidup penyimpanan Anda.

> **Note**
>
> Akun AWSyang membuat bucket memilikinya dan merupakan satu-satunya yang dapat membuat, mengaktifkan, menonaktifkan, atau menghapus aturan siklus hidup.

Untuk membuat dan mengelola siklus aktif untuk bucket S3 di Outposts, lihat topik berikut.

Topik

- Membuat dan mengelola aturan siklus hidup dengan menggunakanAWS Management Console
- Membuat dan mengelola konfigurasi siklus hidup dengan menggunakanAWS CLI dan SDK for Java

Membuat dan mengelola aturan siklus hidup dengan menggunakanAWS Management Console

Anda dapat menggunakan S3 Lifecycle untuk mengoptimalkan kapasitas penyimpanan untuk Amazon S3 di Outposts. Anda dapat membuat aturan siklus hidup untuk kedaluwarsa objek seiring bertambahnya usia atau diganti dengan versi yang lebih baru. Anda dapat membuat, mengaktifkan, menonaktifkan, atau menghapus aturan siklus hidup.

Untuk informasi lebih lanjut tentang S3 Lifecycle, lihatMengelola siklus hidup penyimpanan Anda.

> **Note**
>
> Akun AWSyang membuat bucket memilikinya dan merupakan satu-satunya yang dapat membuat, menonaktifkan, atau menghapus aturan siklus hidup.

Untuk membuat dan mengelola aturan siklus hidup untuk S3 di Outposts dengan menggunakanAWS Management Console, lihat topik berikut.

Topik

- Membuat aturan siklus hidup
- Mengaktifkan aturan siklus hidup
- Mengedit aturan siklus hidup
- Menghapus aturan siklus hidup

Membuat aturan siklus hidup

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi di sebelah kiri, pilih Outposts bucket.
3. Pilih bucket Outposts yang ingin Anda buatkan aturan siklus hidupnya.

4. Pilih tab Manajemen, dan kemudian pilih Buat aturan siklus hidup.
5. Masukkan nilai untuk nama aturan Siklus Hidup.
6. Di bawah Aturan cakupan, pilih salah satu opsi berikut:
   - Untuk membatasi cakupan ke filter tertentu, pilih Batasi cakupan aturan ini menggunakan satu atau beberapa filter. Kemudian, tambahkan filter awalan, tag, atau ukuran objek.
   - Untuk menerapkan aturan pada semua objek dalam bucket, pilih Apply to all object dalam bucket.
7. Di Bawah Tindakan aturan siklus hidup, pilih salah satu opsi berikut:
   - Kedaluwarsa versi objek saat ini - Untuk bucket yang diaktifkan versi, S3 di Outposts menambahkan penanda hapus dan mempertahankan objek sebagai versi noncurrent. Untuk bucket yang tidak menggunakan Versi S3, S3 on Outposts menghapus objek secara permanen.
   - Hapus secara permanen versi objek noncurrent - S3 on Outposts secara permanen menghapus versi objek noncurrent.
   - Hapus penanda penghapusan objek kedaluwarsa atau unggahan multibagian yang tidak lengkap - S3 di Outposts secara permanen menghapus penanda penghapusan objek kedaluwarsa atau unggahan multibagian yang tidak lengkap.

     Jika Anda membatasi cakupan aturan Siklus Hidup dengan menggunakan tag objek, Anda tidak dapat memilih Hapus penanda hapus objek yang kedaluwarsa. Anda juga tidak dapat memilih Hapus penanda hapus objek kedaluwarsa jika Anda memilih Kedaluwarsa versi objek saat ini.

   > **Note**
   >
   > Filter berbasis ukuran tidak dapat digunakan dengan penanda hapus dan unggahan multibagian yang tidak lengkap.

8. Jika Anda memilih Kedaluwarsa versi objek saat ini atau Hapus objek versi noncurrent secara permanen, konfigurasikan pemicu aturan berdasarkan tanggal tertentu atau usia objek.
9. Jika Anda memilih Hapus berakhir objek menghapus penanda, untuk mengkonfirmasi bahwa Anda ingin menghapus berakhir objek menghapus penanda, pilih Hapus berakhir objek menghapus penanda.
10. Di bawah Ringkasan Timeline, tinjau aturan Siklus Hidup Anda, lalu pilih Buat aturan.

Mengaktifkan aturan siklus hidup

Untuk mengaktifkan atau menonaktifkan aturan siklus hidup bucket

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi di sebelah kiri, pilih Outposts bucket.
3. Pilih bucket Outposts yang ingin Anda aktifkan atau nonaktifkan aturan siklus hidupnya.
4. Pilih tab Manajemen, dan kemudian di bawah Aturan siklus hidup, pilih aturan yang ingin Anda aktifkan atau nonaktifkan.
5. Untuk Tindakan, pilih Mengaktifkan atau menonaktifkan aturan.

Mengedit aturan siklus hidup

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi di sebelah kiri, pilih Outposts bucket.
3. Pilih bucket Outposts yang ingin Anda edit aturan siklus hidupnya.
4. Pilih tab Manajemen, dan kemudian pilih Aturan siklus hidup yang ingin Anda edit.
5. (Opsional) Memperbarui nilai untuk nama aturan Siklus Hidup.
6. Di bawah lingkup Aturan, edit cakupan sesuai kebutuhan:
   - Untuk membatasi cakupan ke filter tertentu, pilih Batasi cakupan aturan ini menggunakan satu atau beberapa filter. Kemudian, tambahkan filter awalan, tag, atau ukuran objek.
   - Untuk menerapkan aturan pada semua objek dalam bucket, pilih Apply to all object dalam bucket.
7. Di Bawah Tindakan aturan siklus hidup, pilih salah satu opsi berikut:
   - Kedaluwarsa versi objek saat ini - Untuk bucket yang diaktifkan versi, S3 di Outposts menambahkan penanda hapus dan mempertahankan objek sebagai versi noncurrent. Untuk bucket yang tidak menggunakan Versi S3, S3 on Outposts menghapus objek secara permanen.
   - Hapus secara permanen versi objek noncurrent - S3 on Outposts secara permanen menghapus versi objek noncurrent.
   - Hapus penanda penghapusan objek kedaluwarsa atau unggahan multibagian yang tidak lengkap - S3 di Outposts secara permanen menghapus penanda penghapusan objek kedaluwarsa atau unggahan multibagian yang tidak lengkap.

Jika Anda membatasi cakupan aturan Siklus Hidup dengan menggunakan tag objek, Anda tidak dapat memilih Hapus penanda hapus objek yang kedaluwarsa. Anda juga tidak dapat memilih Hapus penanda hapus objek kedaluwarsa jika Anda memilih Kedaluwarsa versi objek saat ini.

> **Note**
>
> Filter berbasis ukuran tidak dapat digunakan dengan penanda hapus dan unggahan multibagian yang tidak lengkap.

8. Jika Anda memilih Kedaluwarsa versi objek saat ini atau Hapus objek versi noncurrent secara permanen, konfigurasikan pemicu aturan berdasarkan tanggal tertentu atau usia objek.
9. Jika Anda memilih Hapus berakhir objek menghapus penanda, untuk mengkonfirmasi bahwa Anda ingin menghapus berakhir objek menghapus penanda, pilih Hapus berakhir objek menghapus penanda.
10. Pilih Save (Simpan).

Menghapus aturan siklus hidup

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi di sebelah kiri, pilih Outposts bucket.
3. Pilih bucket Outposts yang ingin Anda hapus aturan siklus hidupnya.
4. Pilih tab Manajemen, dan kemudian di bawah Aturan siklus hidup, pilih aturan yang ingin Anda hapus.
5. Pilih Hapus.

Membuat dan mengelola konfigurasi siklus hidup dengan menggunakanAWS CLI dan SDK for Java

Anda dapat menggunakan S3 Lifecycle untuk mengoptimalkan kapasitas penyimpanan untuk Amazon S3 di Outposts. Anda dapat membuat aturan siklus hidup untuk kedaluwarsa objek seiring bertambahnya usia atau diganti dengan versi yang lebih baru. Anda dapat membuat, mengaktifkan, menonaktifkan, menghapus aturan siklus hidup.

Untuk informasi selengkapnya tentang S3 Lifecycle, lihatMengelola siklus hidup penyimpanan Anda.

> **Note**
>
> Akun AWSyang membuat bucket memilikinya dan merupakan satu-satunya yang dapat membuat, mengaktifkan, menonaktifkan, menghapus aturan siklus hidup.

Untuk membuat dan mengelola konfigurasi siklus hidup untuk bucket S3 on Outposts dengan menggunakanAWS Command Line Interface (AWS CLI) danAWS SDK for Java, lihat contoh berikut.

Topik

- Menempatkan sebuah konfigurasi siklus hidup
- DAPATKAN konfigurasi siklus hidup pada bucket Outposts

Menempatkan sebuah konfigurasi siklus hidup

AWS CLI

AWS CLIContoh berikut menempatkan kebijakan konfigurasi siklus hidup pada bucket Outposts. Kebijakan ini menetapkan bahwa semua objek yang memiliki prefiks dan label yang ditandai kedaluwarsa setelah 10 hari.*myprefix* Untuk menggunakan contoh ini, ganti masing-masing*user input placeholder* dengan informasi Anda sendiri.

1. Simpan kebijakan konfigurasi siklus aktif ke berkas JSON. Dalam contoh ini, file diberi nama`lifecycle1.json`.

```
{
    "Rules": [
        {
            "ID": "id-1",
            "Filter": {
                "And": {
                    "Prefix": "myprefix",
                    "Tags": [
                        {
                            "Value": "mytagvalue1",
                            "Key": "mytagkey1"
                        },
                        {
                            "Value": "mytagvalue2",
                            "Key": "mytagkey2"
```

```
                        }
                    ],
                    "ObjectSizeGreaterThan": 1000,
                    "ObjectSizeLessThan": 5000
                }
            },
            "Status": "Enabled",
            "Expiration": {
                "Days": 10
            }
        }
    ]
}
```

2. Kirim file JSON sebagai bagian dari perintah`put-bucket-lifecycle-configuration` CLI. Untuk menggunakan perintah ini, ganti masing-masing*user input placeholder* dengan informasi Anda sendiri. Untuk informasi lebih lanjut tentang perintah ini, lihat put-bucket-lifecycle-configurationdi AWS CLIReferensi.

```
aws s3control put-bucket-lifecycle-configuration --account-id 123456789012 --
bucket arn:aws:s3-outposts:region:123456789012:outpost/op-01ac5d28a6a232904/
bucket/example-outposts-bucket --lifecycle-configuration file://lifecycle1.json
```

SDK for Java

Contoh SDK berikut untuk Java berikut menempatkan sebuah konfigurasi Outposts. konfigurasi siklus hidup.*myprefix* Untuk menggunakan contoh ini, ganti masing-masing*user input placeholder* dengan informasi Anda sendiri. Untuk informasi lebih lanjut, lihat PutBucketLifecycleConfiguration dalam Referensi API Layanan Penyimpanan Sederhana Amazon.

```
import com.amazonaws.services.s3control.model.*;

public void putBucketLifecycleConfiguration(String bucketArn) {

    S3Tag tag1 = new S3Tag().withKey("mytagkey1").withValue("mytagkey1");
    S3Tag tag2 = new S3Tag().withKey("mytagkey2").withValue("mytagkey2");

    LifecycleRuleFilter lifecycleRuleFilter = new LifecycleRuleFilter()
            .withAnd(new LifecycleRuleAndOperator()
                    .withPrefix("myprefix")
```

```
                    .withTags(tag1, tag2))
                    .withObjectSizeGreaterThan(1000)
                    .withObjectSizeLessThan(5000);

    LifecycleExpiration lifecycleExpiration = new LifecycleExpiration()
            .withExpiredObjectDeleteMarker(false)
            .withDays(10);

    LifecycleRule lifecycleRule = new LifecycleRule()
            .withStatus("Enabled")
            .withFilter(lifecycleRuleFilter)
            .withExpiration(lifecycleExpiration)
            .withID("id-1");

    LifecycleConfiguration lifecycleConfiguration = new LifecycleConfiguration()
            .withRules(lifecycleRule);

    PutBucketLifecycleConfigurationRequest reqPutBucketLifecycle = new
 PutBucketLifecycleConfigurationRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn)
            .withLifecycleConfiguration(lifecycleConfiguration);

    PutBucketLifecycleConfigurationResult respPutBucketLifecycle =
 s3ControlClient.putBucketLifecycleConfiguration(reqPutBucketLifecycle);
    System.out.printf("PutBucketLifecycleConfiguration Response: %s%n",
 respPutBucketLifecycle.toString());

}
```

DAPATKAN konfigurasi siklus hidup pada bucket Outposts

AWS CLI

AWS CLIContoh berikut mendapatkan sebuah konfigurasi siklus hidup pada bucket Outposts. Untuk menggunakan perintah ini, ganti masing-masing*user input placeholder* dengan informasi Anda sendiri. Untuk informasi lebih lanjut tentang perintah ini, lihat get-bucket-lifecycle-configurationdi AWS CLIReferensi.

```
aws s3control get-bucket-lifecycle-configuration --account-id 123456789012 --bucket
 arn:aws:s3-outposts:<your-region>:123456789012:outpost/op-01ac5d28a6a232904/
bucket/example-outposts-bucket
```

SDK for Java

Contoh SDK for Java berikut mendapatkan sebuah konfigurasi siklus hidup untuk bucket Outposts. Untuk informasi lebih lanjut, lihat GetBucketLifecycleConfiguration dalam Referensi API Layanan Penyimpanan Sederhana Amazon.

```
import com.amazonaws.services.s3control.model.*;

public void getBucketLifecycleConfiguration(String bucketArn) {

    GetBucketLifecycleConfigurationRequest reqGetBucketLifecycle = new
 GetBucketLifecycleConfigurationRequest()
            .withAccountId(AccountId)
            .withBucket(bucketArn);

    GetBucketLifecycleConfigurationResult respGetBucketLifecycle =
 s3ControlClient.getBucketLifecycleConfiguration(reqGetBucketLifecycle);
    System.out.printf("GetBucketLifecycleConfiguration Response: %s%n",
 respGetBucketLifecycle.toString());

}
```

Replikasi objek untuk S3 di Outposts

Dengan Replikasi S3 aktifAWS Outposts, Anda dapat mengonfigurasi Amazon S3 di Outposts untuk secara otomatis mereplikasi objek S3 di Outposts yang berbeda, atau di antara bucket di Outpost yang sama. Anda dapat menggunakan Replikasi S3 di Outposts untuk mempertahankan beberapa replika data Anda di Outposts yang sama atau berbeda, atau di berbagai akun, untuk membantu memenuhi kebutuhan residensi data. Replikasi S3 di Outposts membantu mendukung kebutuhan penyimpanan Anda yang sesuai dan berbagi data di seluruh akun. Jika Anda perlu memastikan bahwa replika Anda identik dengan data sumber, Anda dapat menggunakan S3 Replication on Outposts untuk membuat replika objek Anda yang menyimpan semua metadata, seperti IDs waktu, tag, dan versi pembuatan objek asli.

Replikasi S3 pada Outposts juga menyediakan metrik dan notifikasi terperinci untuk memantau status replikasi objek antar bucket. Anda dapat menggunakan Amazon CloudWatch untuk memantau kemajuan replikasi dengan melacak replikasi byte yang tertunda, replikasi operasi yang tertunda, dan latensi replikasi antara bucket sumber dan tujuan Anda. Untuk mendiagnosis dan memperbaiki masalah konfigurasi dengan cepat, Anda juga dapat mengatur Amazon EventBridge untuk menerima pemberitahuan tentang kegagalan objek replikasi. Untuk mempelajari selengkapnya, lihat Mengelola replikasi Anda.

Topik

- Konfigurasi Replikasi
- Persyaratan untuk Replication S3 di Outposts
- Apa yang direplikasi?
- Apa yang tidak direplikasi?
- Apa yang tidak didukung oleh Replikasi S3 di Outposts?
- Menyiapkan replikasi
- Mengelola replikasi Anda

Konfigurasi Replikasi

S3 di Outposts menyimpan konfigurasi replikasi sebagai XMLs. Dalam file XML konfigurasi replikasi, Anda menentukan peran AWS Identity and Access Management (IAM) dan satu atau lebih aturan.

```
<ReplicationConfiguration>
    <Role>IAM-role-ARN</Role>
    <Rule>
        ...
    </Rule>
    <Rule>
         ...
    </Rule>
     ...
</ReplicationConfiguration>
```

S3 di Outposts tidak dapat mereplikasi objek tanpa izin Anda. Anda memberikan izin S3 di Outposts dengan peran IAM yang Anda tentukan dalam konfigurasi replikasi. S3 di Outposts mengasumsikan bahwa peran IAM untuk mereplikasi objek atas nama Anda. Anda harus memberikan izin yang

diperlukan untuk peran IAM sebelum memulai replikasi. Untuk informasi lebih lanjut tentang izin ini untuk S3 di Outposts, lihatMembuat peran IAM.

Anda menambahkan satu aturan dalam konfigurasi replikasi dalam skenario berikut:

- Anda ingin mereplikasi semua objek.
- Anda ingin mereplikasi satu subset objek. Anda mengidentifikasi subset objek dengan menambahkan filter pada aturan. Dalam filter, Anda menentukan awalan kunci objek, tag, atau kombinasi keduanya, untuk mengidentifikasi bagian objek yang diterapkan aturan.

Anda menambahkan beberapa aturan dalam konfigurasi replikasi jika Anda ingin mereplikasi subset objek yang berbeda. Dalam setiap aturan, Anda menetapkan filter yang memilih subset objek yang berbeda. Misalnya, Anda dapat memilih untuk mereplikasi objek yang memiliki awalan`document/` kunci`tax/` atau kunci. Untuk melakukan ini, Anda menambahkan dua aturan, satu yang menentukan filter`tax/` key prefix dan satu lagi yang menentukan`document/` key prefix.

Untuk informasi selengkapnya tentang konfigurasi replikasi S3 pada Outposts dan aturan replikasi, lihat ReplicationConfiguration di Referensi API Amazon Simple Storage Service.

Persyaratan untuk Replication S3 di Outposts

Replikasi memerlukan hal berikut:

- Tujuan Outpost kisaran CIDR harus dikaitkan dalam tabel subnet sumber Outpost Anda. Untuk informasi selengkapnya, lihat Prasyarat untuk membuat aturan replikasi.
- Bucket sumber dan tujuan harus memiliki Versi S3 yang diaktifkan. Untuk informasi lebih lanjut tentang versioning, lihat Mengelola versioning.
- Amazon S3 on Outposts harus memiliki izin untuk mereplikasi objek dari bucket sumber ke bucket tujuan atas nama Anda. Itu berarti Anda harus membuat peran layanan untuk mendelegasikan`GET` dan`PUT` izin ke S3 di Outposts.
  1. Sebelum membuat peran layanan, Anda harus memiliki`GET` izin pada bucket sumber dan`PUT` izin pada bucket tujuan.
  2. Untuk membuat peran layanan untuk mendelegasikan izin ke S3 di Outposts, Anda harus terlebih dahulu mengkonfigurasi izin untuk mengizinkan entitas IAM (pengguna atau peran) untuk melakukan`iam:CreateRole` dan`iam:PassRole` tindakan. Kemudian, Anda mengizinkan entitas IAM untuk membuat peran layanan. Untuk membuat S3 di Outposts mengambil peran layanan atas nama Anda dan mendelegasikan`GET` serta`PUT` izin ke S3 di

Outposts, Anda harus menetapkan kebijakan kepercayaan dan izin yang diperlukan untuk peran tersebut. Untuk informasi lebih lanjut tentang izin ini untuk S3 di Outposts, lihatMembuat peran IAM. Untuk informasi selengkapnya tentang pembuatan peran layanan, lihat Membuat peran layanan.

Apa yang direplikasi?

Secara default, S3 di Outposts mereplikasi hal-hal berikut:

- Objek yang dibuat setelah Anda menambahkan konfigurasi replikasi.
- Metadata objek dari objek sumber ke replika. Untuk informasi tentang cara mereplikasi metadata dari replika ke objek sumber, lihatStatus replikasi jika sinkronisasi modifikasi replika Amazon S3 diaktifkan.
- Tag objek, jika ada.

Bagaimana cara menghapus operasi yang memengaruhi replikasi

Jika Anda menghapus objek dari bucket sumber, tindakan berikut terjadi secara default:

- Jika Anda membuat`DELETE` permintaan tanpa menentukan ID versi objek, S3 di Outposts menambahkan penanda hapus. S3 di Outposts berkaitan dengan penanda hapus sebagai berikut:
  - S3 di Outposts tidak mereplikasi penanda hapus secara default.
  - Namun, Anda dapat menambahkan replikasi penanda hapus ke non-tag-based aturan. Untuk informasi selengkapnya tentang cara mengaktifkan replikasi delete marker dalam konfigurasi replikasi Anda, lihatMenggunakan konsol S3.
- Jika Anda menentukan ID versi objek yang akan dihapus dalam`DELETE` permintaan, S3 di Outposts menghapus versi objek tersebut dalam bucket sumber. Namun, itu tidak mereplikasi penghapusan di bucket tujuan. Dengan kata lain, itu tidak menghapus versi objek yang sama dari bucket tujuan. Perilaku ini melindungi data dari penghapusan berbahaya.

Apa yang tidak direplikasi?

Secara default, S3 di Outposts tidak mereplikasi hal-hal berikut:

- Objek dalam bucket sumber yang merupakan replika yang dibuat oleh aturan replikasi lain. Misalnya, Anda mengonfigurasi replikasi di mana bucket A adalah sumbernya dan bucket B adalah tujuannya. Sekarang bayangkan bahwa Anda menambahkan konfigurasi replikasi lain dengan

bucket B sebagai sumber dan bucket C sebagai tujuan. Dalam hal ini, objek dalam bucket B yang merupakan replika objek dalam bucket A tidak direplikasi ke bucket C.

- Objek dalam bucket sumber yang telah direplikasi ke tujuan yang berbeda. Misalnya, jika Anda mengubah bucket tujuan dalam konfigurasi replikasi yang ada, S3 di Outposts tidak akan mereplikasi objek tersebut lagi.
- Objek yang dibuat dengan enkripsi sisi server dengan kunci enkripsi yang disediakan pelanggan (SSE-C).
- Pembaruan untuk subsumber daya tingkat bucket.

  Misalnya, jika Anda mengubah konfigurasi siklus aktif atau menambahkan konfigurasi pemberitahuan ke bucket sumber Anda, perubahan ini tidak diterapkan ke bucket tujuan. Fitur ini memungkinkan untuk memiliki konfigurasi berbeda pada bucket sumber dan tujuan.
- Tindakan yang dilakukan berdasarkan konfigurasi siklus aktif.

  Misalnya, jika Anda mengaktifkan konfigurasi siklus hidup hanya pada bucket sumber Anda dan mengonfigurasi tindakan kedaluwarsa, S3 di Outposts membuat penanda hapus untuk objek yang kedaluwarsa di bucket sumber tetapi tidak mereplikasi penanda tersebut ke bucket tujuan. Jika Anda ingin konfigurasi siklus hidup yang sama diterapkan pada bucket sumber dan tujuan, aktifkan konfigurasi siklus hidup yang sama pada keduanya. Untuk informasi lebih lanjut tentang konfigurasi siklus aktif, lihat Mengelola siklus hidup penyimpanan Anda.

Apa yang tidak didukung oleh Replikasi S3 di Outposts?

Fitur Replication S3 berikut saat ini tidak didukung oleh S3 di Outposts:

- S3 Replication Time Control (S3 RTC). S3 RTC tidak didukung karena lalu lintas objek di Replikasi S3 di Outposts berjalan melalui jaringan lokal Anda (gateway lokal). Untuk informasi selengkapnya tentang gateway lokal, lihat Bekerja dengan gateway lokal di PanduanAWS Outposts Pengguna.
- Replikasi S3 untuk Operasi Batch.

Menyiapkan replikasi

> **Note**
>
> Objek yang ada di bucket sebelum menyiapkan aturan replikasi tidak direplikasi secara otomatis. Dengan kata lain, Amazon S3 di Outposts. tidak mereplikasi objek secara

retroaktif. Untuk mereplikasi objek yang dibuat sebelum konfigurasi replikasi Anda, Anda dapat menggunakan operasi`CopyObject` API untuk menyalinnya ke bucket yang sama. Setelah objek disalin, mereka muncul sebagai objek "baru" di bucket dan konfigurasi replikasi akan berlaku untuk mereka. Untuk informasi selengkapnya tentang menyalin objek, lihatMenyalin objek di Amazon S3 pada bucket Outposts menggunakanAWS SDK for Java dan CopyObjectdi Amazon Simple Storage Service API Reference.

Untuk mengaktifkan Replikasi S3 di Outposts, tambahkan aturan replikasi ke bucket Outposts sumber Anda. Aturan replikasi memberi tahu S3 di Outposts untuk mereplikasi objek seperti yang ditentukan. Dalam aturan replikasi, Anda harus memberikan yang berikut ini:

- Titik akses bucket Outposts sumber - Titik akses Amazon Resource Name (ARN) atau titik akses alias bucket tempat Anda ingin S3 di Outposts mereplikasi objek. Untuk informasi selengkapnya tentang menggunakan alias titik akses, lihat Menggunakan alias gaya ember untuk titik akses bucket S3 Anda di Outposts.
- Objek yang ingin Anda replikasi — Anda dapat mereplikasi semua objek dalam bucket Outposts atau subset. Anda mengidentifikasi bagian dengan menyediakan awalan nama kunci, satu atau lebih tag objek, atau keduanya dalam konfigurasi.

  Misalnya, jika Anda mengonfigurasi aturan replikasi untuk hanya mereplikasi objek dengan awalan nama kunci`Tax/`, S3 pada Outposts mereplikasi objek dengan kunci seperti`Tax/doc1` atau`Tax/doc2`. Tapi itu tidak mereplikasi objek dengan kunci`Legal/doc3`. Jika Anda menentukan prefiks dan satu atau lebih tag, S3 pada Outposts hanya mereplikasi objek yang memiliki key prefix dan tag tertentu.
- Bucket Outposts tujuan - ARN atau titik akses alias ember yang Anda inginkan S3 di Outposts untuk mereplikasi objek.

Anda dapat mengonfigurasi aturan replikasi dengan menggunakan konsol REST API,AWS SDK,AWS Command Line Interface (AWS CLI), atau Amazon S3.

S3 di Outposts juga menyediakan operasi API untuk mendukung pengaturan aturan replikasi. Untuk informasi selengkapnya, lihat topik berikut di Referensi API Amazon Simple Storage Service:

- PutBucketReplication
- GetBucketReplication
- DeleteBucketReplication

Topik

- Prasyarat untuk membuat aturan replikasi
- Membuat aturan replikasi di Outposts

Prasyarat untuk membuat aturan replikasi

Topik

- Menghubungkan subnet Outpost sumber dan tujuan Anda
- Membuat peran IAM

Menghubungkan subnet Outpost sumber dan tujuan Anda

Untuk memiliki lalu lintas replikasi Anda pergi dari Outpost sumber Anda ke Outpost tujuan Anda melalui gateway lokal Anda, Anda harus menambahkan rute baru untuk mengatur jaringan. Anda harus menghubungkan rentang jaringan Classless Inter-Domain Routing (CIDR) dari jalur akses Anda secara bersamaan. Untuk setiap pasangan titik akses, Anda perlu mengatur koneksi ini hanya sekali.

Beberapa langkah untuk mengatur koneksi berbeda, tergantung pada jenis akses titik akhir Outposts Anda yang terkait dengan titik akses Anda. Jenis akses untuk endpoint adalah Private (routing cloud pribadi virtual langsung [VPC] untukAWS Outposts) atau IP milik Pelanggan (kumpulan alamat IP milik pelanggan [CoIP pool] dalam jaringan lokal Anda).

Langkah 1: Temukan rentang CIDR dari endpoint Outposts sumber Anda

Untuk menemukan rentang CIDR titik akhir sumber Anda yang terkait dengan titik akses sumber Anda

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi sebelah kiri, pilih bucket Outposts.
3. Di daftar bucket Outposts, pilih bucket sumber yang Anda inginkan untuk replikasi.
4. Pilih tab Outposts access point, dan pilih Outposts access point untuk source bucket untuk aturan replikasi Anda.
5. Pilih titik akhir Outposts.
6. Salin ID subnet untuk digunakan pada Langkah 5.

7. Metode yang Anda gunakan untuk menemukan rentang CIDR dari titik akhir Outposts sumber tergantung pada jenis akses titik akhir Anda.

   Di bagian Ringkasan endpoint Outposts, lihat Jenis Akses.

   - Jika jenis aksesnya adalah Private, salin nilai Classless inter-domain routing (CIDR) untuk digunakan pada Langkah 6.
   - Jika jenis akses adalah IP Milik Pelanggan, lakukan hal berikut:
     1. Salin nilai pool IPv4 milik Pelanggan untuk digunakan sebagai ID dari address pool nanti.
     2. Buka konsol AWS Outposts di https://console.aws.amazon.com/outposts/.
     3. Di panel navigasi, pilih Tabel rute gateway lokal.
     4. Pilih nilai ID tabel rute gateway lokal dari Outpost sumber Anda.
     5. Di panel rincian, pilih tab CoIP pool. Tempel nilai ID kolam CoIP yang Anda salin sebelumnya di kotak pencarian.
     6. Untuk pool CoIP yang cocok, salin nilai CIDRs yang sesuai dari endpoint Outposts sumber Anda untuk digunakan pada Langkah 6.

Langkah 2: Temukan subnet ID dan rentang CIDR dari titik akhir Outposts tujuan Anda

Untuk menemukan subnet ID dan rentang CIDR dari titik akhir tujuan Anda yang terkait dengan titik akses tujuan Anda, ikuti langkah yang sama di Langkah 1 dan ubah titik akhir Outposts sumber Anda ke titik akhir Outposts tujuan Anda saat Anda menerapkan substeps tersebut. Salin nilai ID subnet dari titik akhir Outposts tujuan Anda untuk digunakan pada Langkah 6. Salin nilai CIDR dari titik akhir Outposts tujuan Anda untuk digunakan pada Langkah 5.

Langkah 3: Temukan ID gateway lokal dari Outpost sumber Anda

Untuk menemukan ID gateway lokal dari Outpost sumber Anda

1. Buka konsol AWS Outposts di https://console.aws.amazon.com/outposts/.
2. Di panel navigasi sebelah kiri, pilih Gateway lokal.
3. Pada halaman gateway lokal, temukan Outpost ID dari Outpost sumber Anda yang ingin Anda gunakan untuk replikasi.
4. Salin nilai ID gateway lokal dari Outpost sumber Anda untuk digunakan pada Langkah 5.

Untuk informasi selengkapnya tentang gateway lokal, lihat Gateway lokal di PanduanAWS Outposts Pengguna.

Langkah 4: Temukan ID gateway lokal dari Outpost tujuan Anda

Untuk menemukan ID gateway lokal dari Outpost tujuan Anda, ikuti langkah yang sama di Langkah 3, kecuali cari Outpost ID untuk Outpost tujuan Anda. Salin nilai ID gateway lokal dari Outpost tujuan Anda untuk digunakan pada Langkah 6.

Langkah 5: Mengatur koneksi dari subnet Outpost sumber Anda ke subnet Outpost tujuan Anda

Untuk terhubung dari subnet Outpost sumber Anda ke subnet Outpost tujuan Anda

1. Masuk keAWS Management Console dan buka konsol VPC di https://console.aws.amazon.com/vpc/.
2. Di panel navigasi kiri, pilih Subnets (Subnet).
3. Di kotak pencarian, masukkan ID subnet untuk titik akhir Outposts sumber Anda yang Anda temukan di Langkah 1. Pilih subnet dengan ID subnet yang cocok.
4. Untuk item subnet yang cocok, pilih Nilai tabel rute dari subnet ini.
5. Pada halaman dengan tabel rute yang dipilih, pilih Tindakan, lalu pilih Edit rute.
6. Pada halaman Edit rute, pilih Tambahkan rute.
7. Di bawah Tujuan, masukkan kisaran CIDR titik akhir Outposts tujuan Anda yang Anda temukan di Langkah 2.
8. Di bawah Target, pilih Outpost Local Gateway, dan masukkan ID gateway lokal dari Outpost sumber Anda yang Anda temukan di Langkah 3.
9. Pilih Save changes (Simpan perubahan).
10. Pastikan Status untuk rute Aktif.

Langkah 6: Mengatur koneksi dari subnet Outpost tujuan Anda ke subnet Outpost sumber Anda

1. Masuk keAWS Management Console dan buka konsol VPC di https://console.aws.amazon.com/vpc/.
2. Di panel navigasi kiri, pilih Subnets (Subnet).
3. Di kotak pencarian, masukkan ID subnet untuk titik akhir Outposts tujuan Anda yang Anda temukan di Langkah 2. Pilih subnet dengan ID subnet yang cocok.
4. Untuk item subnet yang cocok, pilih Nilai tabel rute dari subnet ini.

5. Pada halaman dengan tabel rute yang dipilih, pilih Tindakan, lalu pilih Edit rute.
6. Pada halaman Edit rute, pilih Tambahkan rute.
7. Di bawah Tujuan, masukkan kisaran CIDR dari endpoint Outposts sumber Anda yang Anda temukan di Langkah 1.
8. Di bawah Target, pilih Outpost Local Gateway, dan masukkan ID gateway lokal dari Outpost tujuan Anda yang Anda temukan di Langkah 4.
9. Pilih Save changes (Simpan perubahan).
10. Pastikan Status untuk rute Aktif.

Setelah Anda menghubungkan rentang jaringan CIDR dari titik akses sumber dan tujuan Anda, Anda harus membuat peranAWS Identity and Access Management (IAM).

Membuat peran IAM

Secara default, semua sumber daya S3 di Outposts, bucket, objek, dan subsumber terkait—adalah privat, dan hanya pemilik sumber daya yang dapat mengakses sumber daya. S3 di Outposts membutuhkan izin untuk membaca dan mereplikasi objek dari bucket sumber Outposts. Anda memberikan izin ini dengan membuat peran layanan IAM dan menentukan peran tersebut dalam konfigurasi replikasi Anda.

Bagian ini menjelaskan kebijakan kepercayaan dan kebijakan izin minimum yang diperlukan. Contoh panduan penelusuran memberikan step-by-step instruksi untuk membuat peran IAM. Untuk informasi selengkapnya, lihat Membuat aturan replikasi di Outposts. Untuk informasi lebih lanjut tentang peran IAM, lihat Peran IAM dalam Panduan Pengguna IAM.

- Contoh berikut menunjukkan kebijakan kepercayaan, saat Anda mengidentifikasi S3 di Outposts sebagai kepala layanan yang dapat menjalankan peran tersebut.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3-outposts.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
```

```
}
```

- Contoh berikut menunjukkan kebijakan akses, yaitu Anda memberikan izin peran untuk melakukan tugas replikasi atas nama Anda. Saat S3 di Outposts. memegang peran tersebut, S3 di Outposts memegang peran tersebut, S3 di Outposts memiliki izin yang Anda tentukan dalam kebijakan ini. Untuk menggunakan kebijakan ini, ganti*user input placeholders* dengan informasi Anda sendiri. Pastikan untuk menggantinya dengan Outpost ID dari sumber dan tujuan Anda Outposts dan nama-nama ember dan nama titik akses sumber dan tujuan Outposts ember Anda.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3-outposts:GetObjectVersionForReplication",
            "s3-outposts:GetObjectVersionTagging"
         ],
         "Resource":[
            "arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/
bucket/SOURCE-OUTPOSTS-BUCKET/object/*",
            "arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/
accesspoint/SOURCE-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3-outposts:ReplicateObject",
            "s3-outposts:ReplicateDelete"
         ],
         "Resource":[
            "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-OUTPOST-ID/
bucket/DESTINATION-OUTPOSTS-BUCKET/object/*",
            "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-OUTPOST-ID/
accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
         ]
      }
   ]
}
```

Kebijakan akses memberikan izin untuk tindakan berikut:

- `s3-outposts:GetObjectVersionForReplication`- Izin untuk tindakan ini diberikan pada semua objek untuk memungkinkan S3 pada Outposts untuk mendapatkan versi objek tertentu yang terkait dengan setiap objek.

- `s3-outposts:GetObjectVersionTagging`— Izin untuk tindakan ini pada objek di *`SOURCE-OUTPOSTS-BUCKET`*bucket (bucket sumber) memungkinkan S3 di Outposts membaca tag objek untuk replikasi. Untuk informasi selengkapnya, lihat Menambahkan tag untuk bucket S3 di Outposts. Jika S3 di Outposts tidak memiliki izin ini, S3 di Outposts tidak memiliki izin ini, tetapi tidak akan memberi tag objek.
- `s3-outposts:ReplicateObject`dan`s3-outposts:ReplicateDelete` — Izin untuk tindakan ini pada semua objek dalam *`DESTINATION-OUTPOSTS-BUCKET`*bucket (bucket tujuan) memungkinkan S3 di Outposts untuk mereplikasi objek atau penanda hapus ke bucket Outposts tujuan. Untuk informasi tentang penanda hapus, lihat Bagaimana cara menghapus operasi yang memengaruhi replikasi.

> **Note**
>
> - Izin untuk`s3-outposts:ReplicateObject` tindakan pada *`DESTINATION-OUTPOSTS-BUCKET`*bucket (bucket tujuan) juga memungkinkan replikasi tag objek. Oleh karena itu, Anda tidak perlu secara eksplisit memberikan izin untuk`s3-outposts:ReplicateTags` tindakan tersebut.
> - Untuk replikasi lintas-akun, pemilik bucket Outposts tujuan harus memperbarui kebijakan bucket untuk memberikan izin untuk`s3-outposts:ReplicateObject` tindakan pada *`DESTINATION-OUTPOSTS-BUCKET`*. `s3-outposts:ReplicateObject`Tindakan ini memungkinkan S3 pada Outposts untuk mereplikasi objek dan tag objek ke bucket Outposts tujuan.

Untuk daftar tindakan S3 pada Outposts, lihat Tindakan yang didefinisikan oleh S3 di Outposts.

> **Important**
>
> Akun AWS yang memiliki peran IAM harus memiliki izin untuk tindakan yang diberikannya ke IAM role.
> Misalnya, bayangkan bahwa bucket sumber Outposts berisi objek yang dimiliki oleh lainAkun AWS. Pemilik objek harus secara eksplisit memberikanAkun AWS yang memiliki

IAM role izin yang diperlukan melalui kebijakan bucket dan kebijakan jalur akses. Jika tidak, S3 di Outposts tidak dapat mengakses objek, dan replikasi objek gagal.
Izin yang dijelaskan di sini terkait dengan konfigurasi replikasi minimum. Jika Anda memilih untuk menambahkan konfigurasi replikasi opsional, Anda harus memberikan izin tambahan ke S3 di Outposts.

Memberikan izin ketika bucket sumber dan tujuan Outposts dimiliki oleh yang berbedaAkun AWS

Saat bucket Outposts sumber dan tujuan tidak dimiliki oleh akun yang sama, pemilik bucket Outposts tujuan harus memperbarui kebijakan bucket dan titik akses untuk bucket tujuan. Kebijakan ini harus memberikan pemilik bucket Outposts sumber dan izin peran layanan IAM untuk melakukan tindakan replikasi, seperti yang ditunjukkan dalam contoh kebijakan berikut, atau replikasi akan gagal. Dalam contoh kebijakan ini, *DESTINATION-OUTPOSTS-BUCKET*adalah bucket tujuan. Untuk menggunakan contoh kebijakan ini, ganti*user input placeholders* dengan informasi Anda sendiri.

Jika Anda membuat peran layanan IAM secara manual, tetapkan jalur peran sebagai`role/service-role/`, seperti yang ditunjukkan dalam contoh kebijakan berikut. Untuk informasi lebih lanjut, lihat ARN IAM dalam Panduan Pengguna IAM.

```
{
   "Version":"2012-10-17",
   "Id":"PolicyForDestinationBucket",
   "Statement":[
      {
         "Sid":"Permissions on objects",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::SourceBucket-account-ID:role/service-role/source-
account-IAM-role"
         },
         "Action":[
            "s3-outposts:ReplicateDelete",
            "s3-outposts:ReplicateObject"
         ],
         "Resource":[
            "arn:aws:s3-outposts:region:DestinationBucket-account-
ID:outpost/DESTINATION-OUTPOST-ID/bucket/DESTINATION-OUTPOSTS-BUCKET/object/*"
         ]
      }
```

```
   ]
}
```

```
{
"Version":"2012-10-17",
   "Id":"PolicyForDestinationAccessPoint",
   "Statement":[
      {
         "Sid":"Permissions on objects",
         "Effect":"Allow",
         "Principal":{
            "AWS":"arn:aws:iam::SourceBucket-account-ID:role/service-role/source-
account-IAM-role"
         },
         "Action":[
            "s3-outposts:ReplicateDelete",
            "s3-outposts:ReplicateObject"
         ],
         "Resource" :[
            "arn:aws:s3-outposts:region:DestinationBucket-account-
ID:outpost/DESTINATION-OUTPOST-ID/accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT/
object/*"
         ]
      }
   ]
}
```

> **Note**
>
> Jika objek dalam bucket sumber Outposts ditandai, perhatikan hal berikut:
> Jika pemilik bucket Outposts sumber memberikan izin bucket untuk`s3-outposts:GetObjectVersionTagging` dan`s3-outposts:ReplicateTags` tindakan untuk mereplikasi tag objek (melalui peran IAM), Amazon S3 mereplikasi tag beserta objek. Untuk informasi tentang peran IAM, lihat Membuat peran IAM.

Membuat aturan replikasi di Outposts

S3 Replikasi pada Outposts adalah replikasi objek secara otomatis dan asinkron di seluruh bucket di yang sama atau berbedaAWS Outposts. salinan replikasi objek yang baru dibuat dan pembaruan

Outposts bucket sumber Outposts atau bucket tujuan. Untuk informasi selengkapnya, lihat Replikasi objek untuk S3 di Outposts.

> **Note**
>
> Objek yang ada di bucket sumber Outposts sebelum Anda menyiapkan aturan replikasi tidak direplikasi. Dengan kata lain, S3 di Outposts. tidak mereplikasi objek secara retroaktif. Untuk mereplikasi objek yang dibuat sebelum konfigurasi replikasi Anda, Anda dapat menggunakan operasi`CopyObject` API untuk menyalinnya ke bucket yang sama. Setelah objek disalin, mereka muncul sebagai objek "baru" di bucket dan konfigurasi replikasi akan berlaku untuk mereka. Untuk informasi selengkapnya tentang menyalin objek, lihatMenyalin objek di Amazon S3 pada bucket Outposts menggunakanAWS SDK for Java dan CopyObjectdi Amazon Simple Storage Service API Reference.

Saat Anda mengonfigurasi replikasi, Anda menambahkan aturan replikasi ke bucket sumber Outposts. Aturan replikasi menentukan Outposts yang akan direplikasi dan Outposts di mana objek yang direplikasi akan disimpan. Anda dapat membuat aturan untuk mereplikasi semua objek dalam bucket atau subset objek dengan awalan nama kunci tertentu, satu atau beberapa tag objek, atau keduanya. Sebuah tujuan Pos ember bisa di Outpost yang sama sebagai sumber Outposts ember, atau bisa di Outpost yang berbeda.

Untuk aturan replikasi S3 on Outposts, Anda harus menyediakan titik akses bucket sumber Outposts Amazon Resource Name (ARN) dan titik akses bucket Outposts tujuan ARN alih-alih nama bucket sumber dan tujuan Outposts.

Jika Anda menentukan ID versi objek yang akan dihapus, S3 pada Outposts menghapus versi objek tersebut dalam bucket sumber. Tapi itu tidak mereplikasi penghapusan ke bucket Outposts tujuan. Dengan kata lain, itu tidak menghapus versi objek yang sama dari bucket Outposts tujuan. Perilaku ini melindungi data dari penghapusan berbahaya.

Saat Anda menambahkan aturan replikasi ke bucket Outposts, aturan diaktifkan secara default, sehingga aturan mulai bekerja segera setelah Anda menyimpannya.

Dalam contoh ini, Anda menyiapkan replikasi untuk bucket Outposts sumber dan tujuan yang berada di Outposts yang berbeda dan dimiliki oleh yang samaAkun AWS. Contoh disediakan untuk menggunakan konsol Amazon S3, AWS Command Line Interface (AWS CLI), serta AWS SDK for Java dan AWS SDK for .NET. Untuk informasi tentang replikasi S3 lintas akun di Outposts, lihatMemberikan izin ketika bucket sumber dan tujuan Outposts dimiliki oleh yang berbedaAkun AWS.

Untuk prasyarat untuk mengatur S3 pada aturan replikasi Outposts, lihatPrasyarat untuk membuat aturan replikasi.

Menggunakan konsol S3

Ikuti langkah-langkah ini untuk mengonfigurasi aturan replikasi saat bucket Amazon S3 pada Outposts berada di Outpost yang berbeda dari bucket sumber.

Jika bucket tujuan berada di akun yang berbeda dari bucket sumber, Anda harus menambahkan kebijakan bucket ke bucket ke bucket tujuan untuk memberikan pemilik izin akun bucket sumber untuk mereplikasi objek dalam bucket tujuan untuk mereplikasi objek dalam Outposts tujuan. Untuk informasi selengkapnya, lihat Memberikan izin saat bucket sumber dan tujuan dimiliki oleh yang berbeda Akun AWS.

Untuk membuat aturan replikasi

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di daftar Bucket Outposts, pilih nama bucket yang ingin Anda gunakan sebagai bucket sumber.
3. Pilih tab Manajemen, gulir ke bawah ke bagian Aturan replikasi, lalu pilih Buat aturan replikasi.
4. Untuk nama aturan aplikasi ulang, masukkan nama untuk aturan Anda untuk membantu mengidentifikasi aturan tersebut nanti. Nama wajib diisi dan harus unik dalam bucket.
5. Di bawah Status, Diaktifkan dipilih secara default. Aturan yang diaktifkan mulai berfungsi segera setelah Anda menyimpannya. Jika Anda ingin mengaktifkan aturan nanti, pilih Nonaktif.
6. Di bawah Prioritas, nilai prioritas aturan menentukan aturan mana yang akan diterapkan jika ada aturan yang tumpang tindih. Ketika objek dimasukkan dalam cakupan lebih dari satu aturan replikasi, S3 on Outposts menggunakan nilai prioritas ini untuk menghindari konflik. Secara default, aturan baru ditambahkan ke konfigurasi replikasi pada prioritas tertinggi. Semakin tinggi angkanya, semakin tinggi prioritasnya.

   Untuk mengubah prioritas aturan, setelah Anda menyimpan aturan, pilih nama aturan dari daftar aturan replikasi, pilih Tindakan, lalu pilih Edit prioritas.
7. Di bawah bucket sumber, Anda memiliki opsi berikut untuk mengatur sumber replikasi:

   - Untuk mereplikasi seluruh bucket, pilih Terapkan ke semua objek dalam bucket.
   - Untuk menerapkan awalan atau pemfilteran tag ke sumber replikasi, pilih Batasi cakupan aturan ini dengan menggunakan satu atau beberapa filter. Anda dapat menggabungkan awalan dan tag.

- Untuk mereplikasi semua objek yang memiliki awalan yang sama, di bawah Awalan, masukkan awalan di dalam kotak. Menggunakan prefiks filter membatasi replikasi ke semua objek yang memiliki nama yang dimulai dengan string yang sama (misalnya,`pictures`).

  Jika Anda memasukkan awalan yang merupakan nama folder, Anda harus menggunakan/(depan, garis miring) sebagai karakter terakhir (misalnya,`pictures/`).
- Untuk mereplikasi semua objek yang memiliki satu atau lebih tag objek yang sama, pilih Tambahkan tag, lalu masukkan pasangan nilai kunci ke dalam kotak. Untuk menambahkan tag lain, ulangi prosedur. Untuk informasi lebih lanjut tentang pemberian tanda objek, lihat Menambahkan tag untuk bucket S3 di Outposts.

8. Untuk mengakses S3 Anda di Bucket sumber Outposts untuk replikasi, di bawah nama jalur akses Sumber, pilih titik akses yang dilampirkan ke bucket sumber.
9. Di bawah Tujuan, pilih titik akses ARN dari ember Outposts tujuan di mana Anda ingin S3 di Outposts untuk mereplikasi objek. Tujuan Outposts bucket dapat berada di yang sama atau berbedaAkun AWS dengan bucket sumber.

   Jika bucket tujuan berada di akun yang berbeda dari bucket sumber, Anda harus menambahkan kebijakan bucket ke bucket tujuan untuk memberikan pemilik izin akun bucket Outposts untuk mereplikasi objek ke bucket tujuan untuk mereplikasi objek ke bucket tujuan. Untuk informasi selengkapnya, lihat Memberikan izin ketika bucket sumber dan tujuan Outposts dimiliki oleh yang berbedaAkun AWS.

   > **Note**
   >
   > Jika versioning tidak diaktifkan pada bucket Outposts Anda mendapatkan peringatan yang berisi tombol Aktifkan versioning. Pilih tombol ini untuk mengaktifkan versi pada bucket.

10. Siapkan peran layananAWS Identity and Access Management (IAM) yang dapat diasumsikan S3 di Outposts untuk mereplikasi objek atas nama Anda.

    Untuk menyiapkan peran IAM, di bawah peran IAM, lakukan salah satu hal berikut:

    - Agar S3 di Outposts membuat peran IAM baru untuk konfigurasi replikasi Anda, pilih Pilih dari peran IAM yang ada, lalu pilih Buat peran baru. Saat Anda menyimpan aturan, kebijakan baru akan dibuat untuk peran IAM yang sesuai dengan bucket Outposts sumber dan tujuan yang Anda pilih. Kami menyarankan Anda memilih Buat peran baru.

- Anda juga dapat memilih untuk menggunakan peran IAM yang sudah ada. Jika melakukannya, Anda harus memilih peran yang memberi S3 di Outposts izin yang diperlukan untuk replikasi. Jika peran ini tidak memberikan S3 pada Outposts untuk mengikuti aturan replikasi Anda, replikasi gagal.

  Untuk memilih peran yang ada, pilih Pilih dari peran IAM yang ada, lalu pilih peran dari menu tarik-turun. Anda juga dapat memilih Masukkan IAM role dan kemudian masukkan Amazon Resource Name (ARN) dari peran IAM.

> **Important**
>
> Saat Anda menambahkan aturan replikasi ke bucket S3 pada Outposts, Anda harus memiliki`iam:CreateRole` dan`iam:PassRole` izin untuk dapat membuat dan lulus peran IAM yang memberikan S3 pada izin replikasi Outposts. Untuk informasi lebih lanjut, lihat Memberikan izin pengguna untuk meneruskan peran keLayanan AWS di Panduan Pengguna IAM.

11. Semua objek dalam bucket Outposts dienkripsi secara default. Untuk informasi selengkapnya tentang S3 di enkripsi Outposts. lihatEnkripsi data di S3 di Outposts. Hanya objek yang dienkripsi menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3) yang dapat direplikasi. Replikasi objek yang dienkripsi dengan enkripsi sisi server dengan kunciAWS Key Management Service (AWS KMS) (SSE-KMS) atau enkripsi sisi server dengan kunci enkripsi yang disediakan pelanggan (SSE-C) tidak didukung.
12. Saat diperlukan, aktifkan opsi tambahan berikut saat mengatur konfigurasi aturan replikasi:
    - Jika Anda ingin mengaktifkan metrik replikasi Outposts dalam konfigurasi replikasi Anda, pilih Metrik aplikasi ulang. Untuk informasi selengkapnya, lihat Memantau kemajuan dengan metrik replikasi plikasi.
    - Jika Anda ingin mengaktifkan replikasi delete marker dalam konfigurasi replikasi Anda, pilih Hapus delete marker. Untuk informasi selengkapnya, lihat Bagaimana cara menghapus operasi yang memengaruhi replikasi.
    - Jika Anda ingin mereplikasi perubahan metadata yang dibuat pada replika kembali ke objek sumber, pilih Sinkronisasi modifikasi replika. Untuk informasi selengkapnya, lihat Status replikasi jika sinkronisasi modifikasi replika Amazon S3 diaktifkan.
13. Untuk menyelesaikannya, pilih Buat aturan.

Setelah Anda menyimpan aturan, Anda dapat mengedit, mengaktifkan, menonaktifkan, atau menghapus aturan Anda. Untuk melakukannya, buka tab Manajemen untuk bucket Outposts sumber, gulir ke bawah ke bagian Aturan replikasi, pilih aturan Anda, lalu pilih Edit aturan.

Menggunakan AWS CLI

Untuk menggunakanAWS CLI untuk menyiapkan replikasi saat bucket Outposts dimiliki oleh yang samaAkun AWS, Anda melakukan hal berikut:

- Buat sumber dan tujuan Outposts ember.
- Aktifkan versi pada kedua bucket.
- Buat peran IAM yang memberikan S3 pada Outposts izin untuk mereplikasi objek.
- Tambahkan konfigurasi replikasi ke bucket sumber Outposts.

Untuk memverifikasi konfigurasi Anda, Anda mengujinya.

Untuk mengatur replikasi ketika bucket sumber dan tujuan Outposts dimiliki oleh yang samaAkun AWS

1. Atur profil kredensial untuk AWS CLI. Dalam contoh ini, kami menggunakan nama profil `acctA`. Untuk informasi tentang menyetel profil kredensil, lihat Profil yang ditetapkan di PanduanAWS Command Line Interface Pengguna.

   > **Important**
   >
   > Profil yang Anda gunakan untuk latihan ini harus memiliki izin yang diperlukan. Misalnya, dalam konfigurasi replikasi, Anda menentukan peran layanan IAM yang dapat diasumsikan oleh S3 di Outposts. Anda dapat melakukan ini hanya jika profil yang Anda gunakan memiliki`iam:CreateRole` dan`iam:PassRole` izin. Untuk informasi lebih lanjut, lihat Memberikan izin pengguna untuk meneruskan peran keLayanan AWS di Panduan Pengguna IAM. Jika Anda menggunakan kredensil administrator untuk membuat profil yang ditetapkan, profil yang ditetapkan akan memiliki izin yang diperlukan untuk melakukan semua tugas.

2. Buat bucket sumber dan mengaktifkan versioning di dalamnya. `create-bucket`Perintah berikut membuat*`SOURCE-OUTPOSTS-BUCKET`* bucket di Wilayah US East (N. Virginia) (`us-east-1`). Untuk menggunakan perintah ini, ganti*`user input placeholders`* dengan informasi Anda sendiri.

```
aws s3control create-bucket --bucket SOURCE-OUTPOSTS-BUCKET --outpost-id SOURCE-
OUTPOST-ID --profile acctA --region us-east-1
```

`put-bucket-versioning`Perintah berikut memungkinkan versi pada*SOURCE-OUTPOSTS-BUCKET* bucket. Untuk menggunakan perintah ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/bucket/SOURCE-OUTPOSTS-
BUCKET --versioning-configuration Status=Enabled --profile acctA
```

3. Buat bucket tujuan dan mengaktifkan versioning di dalamnya. `create-bucket`Perintah berikut membua*DESTINATION-OUTPOSTS-BUCKET* bucket di Wilayah US West (Oregon) (`us-west-2`). Untuk menggunakan perintah ini, ganti*user input placeholders* dengan informasi Anda sendiri.

> **Note**
>
> Untuk menyiapkan konfigurasi replikasi saat bucket Outposts dan tujuan berada di yang samaAkun AWS, Anda menggunakan profil dengan nama yang sama. Contoh ini menggunakan `acctA`. Untuk menguji konfigurasi replikasi saat bucket dimiliki oleh yang berbedaAkun AWS, Anda menentukan profil yang berbeda untuk setiap bucket.

```
aws s3control create-bucket --bucket DESTINATION-OUTPOSTS-BUCKET --create-bucket-
configuration LocationConstraint=us-west-2 --outpost-id DESTINATION-OUTPOST-ID --
profile acctA --region us-west-2
```

`put-bucket-versioning`Perintah berikut memungkinkan versi pada*DESTINATION-OUTPOSTS-BUCKET* bucket. Untuk menggunakan perintah ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
aws s3control put-bucket-versioning --account-id 123456789012 --bucket arn:aws:s3-
outposts:region:123456789012:outpost/DESTINATION-OUTPOST-ID/bucket/DESTINATION-
OUTPOSTS-BUCKET --versioning-configuration Status=Enabled --profile acctA
```

4. Buat peran layanan IAM . Kemudian dalam konfigurasi replikasi, Anda menambahkan peran layanan ini ke*SOURCE-OUTPOSTS-BUCKET* bucket. S3 di Outposts. mengasumsikan peran ini untuk mereplikasi objek atas nama Anda. Anda membuat peran IAM dalam dua langkah:

   a. Buatlah IAM role.

      i. Salin kebijakan kepercayaan berikut dan simpan di berkas dengan nama `s3-on-outposts-role-trust-policy.json` di direktori saat ini di komputer lokal Anda. Kebijakan ini memberi S3 izin utama layanan Outposts untuk memegang peran layanan.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":"s3-outposts.amazonaws.com"
         },
         "Action":"sts:AssumeRole"
      }
   ]
}
```

      ii. Jalankan perintah berikut untuk membuat peran. Ganti*user input placeholders* dengan informasi Anda sendiri.

```
aws iam create-role --role-name replicationRole --assume-role-policy-
document file://s3-on-outposts-role-trust-policy.json --profile acctA
```

   b. Lampirkan kebijakan izin pada peran layanan.

      i. Salin kebijakan izin berikut dan simpan ke berkas dengan nama `s3-on-outposts-role-permissions-policy.json` di direktori saat ini di komputer lokal Anda. Kebijakan ini memberikan izin untuk berbagai S3 di bucket dan tindakan objek Outposts. Untuk menggunakan kebijakan ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
{
   "Version":"2012-10-17",
   "Statement":[
```

```
      {
         "Effect":"Allow",
         "Action":[
            "s3-outposts:GetObjectVersionForReplication",
            "s3-outposts:GetObjectVersionTagging"
         ],
         "Resource":[
            "arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-
OUTPOST-ID/bucket/SOURCE-OUTPOSTS-BUCKET/object/*",
            "arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-
OUTPOST-ID/accesspoint/SOURCE-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3-outposts:ReplicateObject",
            "s3-outposts:ReplicateDelete"
         ],
         "Resource":[
            "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-
OUTPOST-ID/bucket/DESTINATION-OUTPOSTS-BUCKET/object/*",
            "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-
OUTPOST-ID/accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT/object/*"
         ]
      }
   ]
}
```

ii. Jalankan perintah berikut untuk membuat kebijakan dan melampirkannya ke peran. Ganti*user input placeholders* dengan informasi Anda sendiri.

```
aws iam put-role-policy --role-name replicationRole --policy-
document file://s3-on-outposts-role-permissions-policy.json --policy-
name replicationRolePolicy --profile acctA
```

5. Tambahkan konfigurasi replikasi ke*SOURCE-OUTPOSTS-BUCKET* bucket.

   a. Meskipun S3 on Outposts API memerlukan konfigurasi replikasi dalam format XML,AWS CLI mengharuskan Anda menentukan konfigurasi replikasi dalam format JSON. Simpan JSON berikut dalam berkas yang disebut `replication.json` ke direktori lokal di komputer Anda. Untuk menggunakan konfigurasi ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
{
  "Role": "IAM-role-ARN",
  "Rules": [
    {
      "Status": "Enabled",
      "Priority": 1,
      "DeleteMarkerReplication": { "Status": "Disabled" },
      "Filter" : { "Prefix": "Tax"},
      "Destination": {
        "Bucket":
        "arn:aws:s3-outposts:region:123456789012:outpost/DESTINATION-OUTPOST-
ID/accesspoint/DESTINATION-OUTPOSTS-BUCKET-ACCESS-POINT"
      }
    }
  ]
}
```

b. Jalankan`put-bucket-replication` perintah berikut untuk menambahkan konfigurasi replikasi ke bucket Outposts sumber Anda. Untuk menggunakan perintah ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
aws s3control put-bucket-replication --account-id 123456789012 --
bucket arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-
ID/bucket/SOURCE-OUTPOSTS-BUCKET --replication-configuration file://
replication.json --profile acctA
```

c. Untuk mengambil konfigurasi replikasi, gunakan perintah `get-bucket-replication`. Untuk menggunakan perintah ini, ganti*user input placeholders* dengan informasi Anda sendiri.

```
aws s3control get-bucket-replication --account-id 123456789012 --bucket
 arn:aws:s3-outposts:region:123456789012:outpost/SOURCE-OUTPOST-ID/
bucket/SOURCE-OUTPOSTS-BUCKET --profile acctA
```

6. Uji konfigurasi di konsol Amazon S3:

   a. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

   b. Di*SOURCE-OUTPOSTS-BUCKET* bucket, buat folder bernama`Tax`.

   c. Tambahkan objek sampel ke`Tax` folder di*SOURCE-OUTPOSTS-BUCKET* bucket.

d. Di*DESTINATION-OUTPOSTS-BUCKET* bucket, verifikasi hal berikut:

- S3 di Outposts. mereplikasi objek.

> **Note**
>
> Jumlah waktu yang diperlukan untuk S3 di Outposts untuk mereplikasi suatu objek tergantung pada ukuran objek. Untuk informasi tentang cara melihat status replikasi, lihat Mendapatkan informasi status replikasi.

- Pada tab Properti objek, status Replikasi diatur ke Replika (mengidentifikasi ini sebagai objek replika).

Mengelola replikasi Anda

Bagian ini menjelaskan opsi konfigurasi replikasi tambahan yang tersedia di S3 di Outposts, cara menentukan status replikasi, dan cara memecahkan masalah replikasi. Untuk informasi tentang konfigurasi replikasi inti, lihat Menyiapkan replikasi.

Topik

- Memantau kemajuan dengan metrik replikasi plikasi
- Mendapatkan informasi status replikasi
- Mengatasi masalah replikasi
- Menggunakan EventBridge untuk Replikasi S3 di Outposts

Memantau kemajuan dengan metrik replikasi plikasi

S3 Replication on Outposts memberikan metrik terperinci untuk aturan replikasi dalam konfigurasi replikasi dalam konfigurasi replikasi Anda. Dengan metrik replikasi, Anda dapat memantau kemajuan replikasi dengan melacak replikasi dengan melacak replikasi byte yang menunggu replikasi, replikasi latensi replikasi, dan operasi yang menunggu replikasi. Untuk membantu memecahkan masalah konfigurasi, Anda juga dapat mengatur Amazon EventBridge guna menerima pemberitahuan tentang kegagalan replikasi.

Saat metrik replikasi diaktifkan, S3 Replication on Outposts menerbitkan metrik berikut ke Amazon CloudWatch:

- Bytes Pending Replication - Jumlah total byte objek yang menunggu replikasi untuk aturan replikasi yang diberikan.
- Latensi Replication - Jumlah detik maksimum detik ketika bucket tujuan replikasi berada di belakang bucket sumber untuk aturan replikasi yang diberikan.
- Operations Pending Replication - Jumlah operasi yang menunggu replikasi untuk aturan replikasi yang diberikan. Operasi mencakup objek, delete marker, dan tag.

> **Note**
>
> S3 Replication on Outposts metrik ditagih dengan tarif yang sama dengan metrik CloudWatch khusus. Untuk informasi selengkapnya, lihat Harga CloudWatch.

Mendapatkan informasi status replikasi

Status replikasi dapat membantu Anda menentukan status saat ini objek yang sedang direplikasi oleh Amazon S3 di Outposts. Status replikasi dari objek sumber akan kembali denga `PENDING`, `COMPLETED`, atau `FAILED`. Status replikasi dari replika akan mengembalikan `REPLICA`.

Gambaran status replikasi

Dalam skenario replikasi, Anda memiliki bucket sumber tempat Anda mengonfigurasi replikasi dan bucket tujuan tempat S3 di Outposts mereplikasi objek. Saat Anda meminta objek (using`GetObject`) atau metadata objek (using`HeadObject`) dari bucket ini, S3 on Outposts mengembalikan`x-amz-replication-status` header dalam respons sebagai berikut:

- Saat Anda meminta objek dari bucket sumber, S3 on Outposts mengembalikan`x-amz-replication-status` header jika objek di permintaan Anda memenuhi syarat untuk replikasi.

  Misalnya, Anda menetapkan prefiks objek`TaxDocs` dalam konfigurasi replikasi Anda untuk memberi tahu S3 di Outposts agar hanya mereplikasi objek dengan prefiks nama kunci`TaxDocs`. Objek apa pun yang Anda unggah yang memiliki prefiks nama kunci ini—misalnya, `TaxDocs/document1.pdf`—akan direplikasi. Untuk permintaan objek dengan awalan nama kunci ini, S3 pada Outposts mengembalikan`x-amz-replication-status` header dengan salah satu nilai berikut untuk status replikasi objek:`PENDING``COMPLETED`, atau`FAILED`.

> **Note**
>
> Jika replikasi objek gagal setelah Anda mengunggah sebuah objek, Anda tidak dapat mencoba ulang replikasi. Anda harus mengunggah objek lagi. Objek bertransisi menjadi`FAILED` kondisi karena masalah seperti izin peran replikasi yang hilang atau izin bucket yang hilang. Untuk kegagalan sementara, seperti jika bucket atau pos terdepan Anda tidak tersedia, status replikasi tidak berpindah ke`FAILED`, tetapi tetap ada`PENDING`. Setelah sumber daya kembali online, S3 on Outposts melanjutkan replikasi objek tersebut.

- Saat Anda meminta objek dari bucket tujuan, jika objek dalam permintaan Anda adalah replika yang dibuat S3 on Outposts, S3 on Outposts mengembalikan`x-amz-replication-status` header dengan nilai`REPLICA`.

> **Note**
>
> Sebelum menghapus sebuah objek dari bucket sumber yang memiliki pengaktifan replikasi, periksa status replikasi objek untuk memastikan bahwa objek tersebut telah direplikasi.

Status replikasi jika sinkronisasi modifikasi replika Amazon S3 diaktifkan

Saat aturan replikasi Anda mengaktifkan sinkronisasi modifikasi replika S3, replika dapat melaporkan status selain`REPLICA`. Jika perubahan metadata sedang dalam proses mereplikasi,`x-amz-replication-status` header untuk replika akan kembali`PENDING`. Jika sinkronisasi modifikasi replika gagal mereplikasi metadata, header untuk replika akan kembali`FAILED`. Jika metadata direplikasi dengan benar, header untuk replika mengembalikan nilai`REPLICA`.

Mengatasi masalah replikasi

Jika replika objek tidak muncul di bucket Amazon S3 setelah Anda mengonfigurasi replikasi, gunakan kiat pemecahan masalah ini untuk mengidentifikasi dan memperbaiki masalah.

- Waktu yang dibutuhkan S3 di Outposts untuk mereplikasi objek tergantung pada beberapa faktor, termasuk jarak antara sumber dan Outposts tujuan, dan ukuran objek.

  Anda dapat memeriksa status replikasi objek sumber. Jika status replikasi objek adalah`PENDING`, S3 pada Outposts belum menyelesaikan replikasi. Jika status replikasi objek adalah`FAILED`, periksa konfigurasi replikasi yang Anda tetapkan pada bucket sumber.

- Dalam konfigurasi replikasi pada bucket sumber, verifikasi hal berikut:
  - Titik akses Amazon Resource Name (ARN) bucket tujuan sudah benar.
  - Awalan nama kunci benar. Misalnya, jika Anda mengatur konfigurasi untuk mereplikasi objek dengan awalan `Tax`, maka hanya objek dengan nama kunci seperti `Tax/document1` atau `Tax/document2` direplikasi. Objek dengan nama kunci `document3` tidak direplikasi.
  - Statusnya adalah `Enabled`.
- Verifikasi bahwa versi belum ditangguhkan pada salah satu bucket. Bucket sumber dan tujuan harus memiliki versioning yang diaktifkan.
- Jika bucket tujuan dimiliki oleh Akun AWS lain, verifikasi bahwa pemilik bucket memiliki kebijakan bucket tujuan yang memungkinkan pemilik bucket sumber mereplikasi objek. Sebagai contoh, lihat Memberikan izin ketika bucket sumber dan tujuan Outposts dimiliki oleh yang berbedaAkun AWS.
- Jika replika objek tidak muncul di bucket tujuan, masalah berikut dapat mencegah replikasi:
  - S3 pada Outposts tidak mereplikasi objek dalam bucket sumber yang merupakan replika yang dibuat oleh konfigurasi replikasi lain. Misalnya, jika Anda mengatur konfigurasi replikasi dari bucket A ke bucket B ke bucket C, S3 di Outposts tidak mereplikasi replika objek di bucket B ke bucket C.

    Jika Anda ingin mereplikasi objek dalam bucket A ke bucket B dan bucket C, tetapkan beberapa tujuan bucket dalam aturan replikasi yang berbeda untuk konfigurasi replikasi bucket sumber Anda. Misalnya, buat dua aturan replikasi pada bucket sumber A, dengan satu aturan untuk mereplikasi ke bucket tujuan B dan aturan lainnya untuk mereplikasi ke bucket tujuan C.
  - Pemilik bucket sumber dapat memberikan izin Akun AWS lainnya untuk mengunggah objek. Secara default, pemilik bucket sumber tidak memiliki izin untuk objek yang dibuat oleh akun lain. Konfigurasi replikasi hanya mereplikasi objek yang izin aksesnya dimiliki pemilik bucket sumber. Untuk menghindari masalah replikasi, pemilik bucket sumber dapat memberikanAkun AWS izin lain untuk membuat objek secara kondisional, yang memerlukan izin akses eksplisit pada objek tersebut. Untuk melihat contoh kebijakan IAM, lihat Berikan izin lintas akun untuk unggah objek sekaligus memastikan bahwa pemilik bucket memiliki kendali penuh.
- Misalkan bahwa dalam konfigurasi replikasi, Anda menambahkan aturan untuk mereplikasi subset objek yang memiliki tag tertentu. Dalam kasus ini, Anda harus menetapkan kunci dan nilai tag spesifik pada saat objek dibuat agar S3 di Outposts mereplikasi objek. Jika Anda membuat objek terlebih dahulu lalu menambahkan tag ke objek yang sudah ada, S3 pada Outposts tidak mereplikasi objek.
- Replikasi gagal jika kebijakan bucket menghalangi akses ke peran replikasi untuk tindakan-tindakan berikut:

Bucket sumber:

```
"s3-outposts:GetObjectVersionForReplication",
"s3-outposts:GetObjectVersionTagging"
```

Bucket tujuan:

```
"s3-outposts:ReplicateObject",
"s3-outposts:ReplicateDelete",
"s3-outposts:ReplicateTags"
```

- Amazon EventBridge dapat memberi tahu Anda ketika objek tidak mereplikasi ke Outposts tujuan mereka. Untuk informasi selengkapnya, lihat Menggunakan EventBridge untuk Replikasi S3 di Outposts.

Menggunakan EventBridge untuk Replikasi S3 di Outposts

Amazon S3 di Outposts terintegrasi dengan Amazon EventBridge dan menggunakan`s3-outposts` namespace. EventBridge adalah layanan bus peristiwa yang dapat Anda gunakan untuk menghubungkan aplikasi Anda dengan data dari berbagai sumber. Untuk informasi lebih lanjut, lihat Apa itu Amazon EventBridge? di Panduan EventBridge Pengguna Amazon.

Untuk membantu memecahkan masalah apa pun, Anda dapat mengatur Amazon EventBridge untuk menerima notifikasi tentang peristiwa replikasi. EventBridge dapat memberitahu Anda dalam kasus ketika objek tidak mereplikasi ke Outposts tujuan mereka. Untuk informasi lebih lanjut tentang status saat ini objek yang sedang direplikasi, lihatGambaran status replikasi.

Setiap kali peristiwa tertentu terjadi di ember Outposts Anda, S3 on Outposts dapat mengirim acara ke EventBridge. Tidak seperti tujuan lain, Anda tidak perlu memilih jenis Peristiwa yang ingin Anda kirimkan. Anda juga dapat menggunakan EventBridge aturan untuk merutekan peristiwa ke target tambahan. Setelah EventBridge diaktifkan, S3 on Outposts mengirimkan semua acara berikut ke EventBridge.

Tipe peristiwa	Deskripsi	Namespace
`Operation FailedReplication`	Replikasi objek dalam aturan replikasi gagal. Untuk informasi lebih lanjut tentang S3 Replikasi Outposts Alasan,	`s3-outposts`

Tipe peristiwa	Deskripsi	Namespace
	lihatMenggunakan EventBridge untuk melihat S3 Replikasi pada alasan kegagalan Outposts.	

Menggunakan EventBridge untuk melihat S3 Replikasi pada alasan kegagalan Outposts

Tabel berikut berisi daftar S3 Replikasi pada alasan kegagalan Outposts. Anda dapat mengonfigurasi EventBridge aturan untuk mempublikasikan dan melihat alasan melalui Amazon Simple Queue Service (Amazon SQS), Amazon Simple Notification Service (Amazon SNS)AWS Lambda, atau Amazon CloudWatch Logs. Untuk informasi selengkapnya tentang izin yang diperlukan untuk menggunakan sumber daya ini EventBridge, lihat Menggunakan kebijakan berbasis sumber daya untuk EventBridge.

Alasan	Deskripsi
`AssumeRoleNotPermitted`	S3 di Outposts tidak dapat mengasumsikan peranAWS Identity and Access Managemen t (IAM) yang ditentukan dalam konfigurasi replikasi.
`DstBucketNotFound`	S3 di Outposts tidak dapat menemukan bucket tujuan yang ditentukan dalam konfigurasi replikasi.
`DstBucketUnversioned`	Pembuatan versi tidak diaktifkan pada bucket tujuan Outposts. Untuk mereplikasi objek dengan Replikasi S3 di Outposts, Anda harus mengaktifkan versi pada bucket tujuan.
`DstDelObjNotPermitted`	S3 di Outposts tidak dapat mereplikasi penghapusan ke bucket tujuan. `s3-outpos ts:ReplicateDelete` Izin mungkin hilang untuk bucket tujuan.
`DstMultipartCompleteNotPermitted`	S3 di Outposts. tidak dapat menyelesa ikan unggahan multibagian dari objek di

Alasan	Deskripsi
	ember tujuan. `s3-outposts:ReplicateObject` Izin mungkin hilang untuk bucket tujuan.
`DstMultipartInitNotPermitted`	S3 di Outposts tidak dapat memulai unggahan multibagian dari objek ke ember tujuan. `s3-outposts:ReplicateObject` Izin mungkin hilang untuk bucket tujuan.
`DstMultipartPartUploadNotPermitted`	S3 di Outposts tidak dapat mengunggah objek multipart di bucket tujuan. `s3-outposts:ReplicateObject` Izin mungkin hilang untuk bucket tujuan.
`DstOutOfCapacity`	S3 on Outposts tidak dapat mereplikasi ke Outpost tujuan karena Outpost berada di luar kapasitas penyimpanan S3.
`DstPutObjNotPermitted`	S3 di Outposts tidak dapat mereplikasi objek ke bucket tujuan. `s3-outposts:ReplicateObject` Izin mungkin hilang untuk bucket tujuan.
`DstPutTaggingNotPermitted`	S3 di Outposts tidak dapat mereplikasi tanda objek ke ember tujuan. `s3-outposts:ReplicateObject` Izin mungkin hilang untuk bucket tujuan.
`DstVersionNotFound`	S3 di Outposts tidak dapat menemukan versi objek yang diperlukan di bucket tujuan untuk mereplikasi metadata versi objek tersebut.
`SrcBucketReplicationConfigMissing`	S3 di Outposts tidak dapat menemukan konfigurasi replikasi untuk titik akses yang terkait dengan bucket sumber Outposts.

Alasan	Deskripsi
`SrcGetObjNotPermitted`	S3 di Outposts tidak dapat mengakses objek dalam bucket sumber untuk replikasi . `s3-outposts:GetObjectVersionForReplication` Izin mungkin hilang untuk bucket sumber.
`SrcGetTaggingNotPermitted`	S3 di Outposts tidak dapat mengakses informasi tag objek dari bucket sumber. `s3-outposts:GetObjectVersionTagging` Izin mungkin hilang untuk bucket sumber.
`SrcHeadObjectNotPermitted`	S3 di Outposts tidak dapat mengambil metadata objek dari bucket sumber. `s3-outposts:GetObjectVersionForReplication` Izin mungkin hilang untuk bucket sumber.
`SrcObjectNotEligible`	Objek tidak memenuhi syarat untuk replikasi. Objek atau tag objeknya tidak cocok dengan konfigurasi replikasi.

Untuk informasi lebih lanjut tentang replikasi, lihat topik berikut:

- Membuat peran IAM
- Mengatasi masalah replikasi

Pemantauan EventBridge dengan CloudWatch

Untuk pemantauan, Amazon EventBridge terintegrasi dengan Amazon CloudWatch. EventBridge secara otomatis mengirimkan metrik ke CloudWatch setiap menit. Metrik ini mencakup jumlah peristiwa yang telah dicocokkan dengan aturan dan berapa kali target dipanggil oleh aturan. Saat aturan masuk EventBridge, semua target terkait dengan aturan dipanggil. Anda dapat memantau EventBridge perilaku Anda melalui CloudWatch cara-cara berikut.

- Anda dapat memantau EventBridgemetrik yang tersedia untuk EventBridge aturan Anda dari CloudWatch dasbor. Kemudian, Anda dapat menggunakan CloudWatch fitur, seperti CloudWatch alarm, untuk mengatur alarm pada metrik tertentu. Jika metrik tersebut mencapai nilai ambang batas khusus yang telah Anda tentukan di alarm, Anda menerima notifikasi dan dapat mengambil tindakan yang sesuai.
- Anda dapat menetapkan CloudWatch Log Amazon sebagai target EventBridge aturan Anda. Kemudian, EventBridge buat log stream dan CloudWatch Log menyimpan teks dari peristiwa sebagai entri. Untuk informasi lebih lanjut, lihat EventBridge dan CloudWatch Log.

Untuk informasi lebih lanjut tentang men-debug EventBridge Peristiwa dan pengarsipan Peristiwa, lihat topik berikut:

- Kebijakan dan menggunakan antrean surat
- Pengarsipan EventBridge acara

Berbagi S3 di Outposts dengan menggunakan AWS RAM

Amazon S3 di Outposts mendukung berbagi kapasitas S3 di beberapa akun dalam organisasi dengan menggunakan (). AWS Resource Access Manager AWS RAM Dengan berbagi S3 on Outposts, Anda dapat mengizinkan orang lain membuat dan mengelola bucket, titik akhir, dan titik akses di Outpost Anda.

Topik ini menunjukkan cara menggunakan AWS RAM untuk berbagi S3 di Outposts dan sumber daya terkait dengan yang lain Akun AWS di organisasi Anda. AWS

Prasyarat

- Akun pemilik Outpost memiliki organisasi yang dikonfigurasi. AWS Organizations Untuk informasi selengkapnya, lihat Membuat organisasi di Panduan AWS Organizations Pengguna.
- Organisasi ini menyertakan Akun AWS yang ingin Anda bagikan kapasitas S3 on Outposts Anda. Untuk informasi selengkapnya, lihat Mengirim undangan ke Akun AWS dalam AWS Organizations Panduan Pengguna.
- Pilih salah satu opsi berikut yang ingin Anda bagikan. Sumber daya kedua (baik Subnet atau Outposts) harus dipilih sehingga titik akhir juga dapat diakses. Titik akhir adalah persyaratan jaringan untuk mengakses data yang disimpan di S3 di Outposts.

Opsi 1	Opsi 2
S3 di Outposts Memungkinkan pengguna membuat bucket di Outposts dan titik akses Anda dan menambahkan objek ke bucket tersebut.	S3 di Outposts Memungkinkan pengguna membuat bucket di Outposts dan titik akses Anda dan menambahkan objek ke bucket tersebut.
Subnet Memungkinkan pengguna untuk menggunak an virtual private cloud (VPC) dan titik akhir yang terkait dengan subnet Anda.	Outposts Memungkinkan pengguna untuk melihat bagan kapasitas S3 dan halaman beranda AWS Outposts konsol. Juga memungkinkan pengguna untuk membuat subnet di Outposts bersama dan membuat titik akhir.

Prosedur

1. Masuk ke AWS Management Console dengan menggunakan Akun AWS yang memiliki Outpost, lalu buka AWS RAM konsol di https://console.aws.amazon.com/ram.
2. Pastikan Anda telah mengaktifkan berbagi dengan AWS Organizations in AWS RAM. Untuk selengkapnya, lihat Mengaktifkan berbagi sumber daya AWS Organizations di dalam Panduan AWS RAM Pengguna.
3. Gunakan Opsi 1 atau Opsi 2 dalam prasyarat untuk membuat pembagian sumber daya. Jika Anda memiliki beberapa S3 di sumber daya Outposts, pilih Amazon Resource Names (ARN) dari resource yang ingin Anda bagikan. Untuk mengaktifkan titik akhir, bagikan subnet atau Outpost Anda.

   Untuk informasi selengkapnya tentang cara membuat pembagian sumber daya, lihat Membuat berbagi sumber daya di Panduan AWS RAM Pengguna.
4. Akun AWS Yang Anda bagikan sumber daya Anda sekarang harus dapat menggunakan S3 di Outposts. Bergantung pada opsi yang Anda pilih dalam prasyarat, berikan informasi berikut kepada pengguna akun:

Opsi 1	Opsi 2
Outpost ID ID VPC ID subnet. ID dari grup keamanan	Outpost ID

> **Note**
>
> Pengguna dapat mengonfirmasi bahwa sumber daya telah dibagikan dengan mereka menggunakan AWS RAM konsol, AWS Command Line Interface (AWS CLI), AWS SDK, atau REST API. Pengguna dapat melihat pembagian sumber daya yang ada dengan menggunakan perintah get-resource-sharesCLI.

Contoh penggunaan

Setelah Anda membagikan sumber daya S3 di Outposts dengan akun lain, akun tersebut dapat mengelola bucket dan objek di Outpost Anda. Jika Anda membagikan sumber daya Subnet, maka akun tersebut dapat menggunakan titik akhir yang Anda buat. Contoh berikut menunjukkan bagaimana pengguna dapat menggunakan AWS CLI untuk berinteraksi dengan Outpost Anda setelah Anda membagikan sumber daya ini.

Example Buat bucket

Contoh berikut membuat bucket bernama *example-s3-bucket1* di Outpost. *op-01ac5d28a6a232904* Sebelum menggunakan perintah ini, ganti masing-masing *user input placeholder* dengan nilai yang sesuai untuk kasus penggunaan Anda.

```
aws s3control create-bucket --bucket example-s3-bucket1 --outpost-
id op-01ac5d28a6a232904
```

Untuk informasi selengkapnya tentang perintah ini, lihat create-bucket di Referensi.AWS CLI

Example Membuat Titik Akses

Contoh berikut membuat titik akses pada Outpost dengan menggunakan parameter contoh dalam tabel berikut. Sebelum menggunakan perintah ini, ganti *user input placeholder* nilai-nilai ini dan Wilayah AWS kode dengan nilai yang sesuai untuk kasus penggunaan Anda.

Parameter	Nilai
account-id	*111122223333*
Nama titik akses	*example-outpost-access-point*
outpost-id	*op-01ac5d28a6a232904*
Nama bucket outpost	*example-s3-bucket1*
ID VPC	*vpc-1a2b3c4d5e6f7g8h9*

> **Note**
>
> Parameter ID Akun harus berupa Akun AWS ID pemilik bucket, yang merupakan pengguna bersama.

```
aws s3control create-access-point --account-id 111122223333 --name example-outpost-
access-point \
--bucket arn:aws:s3-outposts:us-east-1:111122223333:outpost/op-01ac5d28a6a232904/
bucket/example-s3-bucket1 \
--vpc-configuration VpcId=vpc-1a2b3c4d5e6f7g8h9
```

Untuk informasi selengkapnya tentang perintah ini, lihat create-access-pointdi AWS CLI Referensi.

Example Mengunggah objek

Contoh berikut mengunggah file *my_image.jpg* dari sistem file lokal pengguna ke objek bernama *images/my_image.jpg* melalui titik akses *example-outpost-access-point* di Outpost *op-01ac5d28a6a232904*, yang dimiliki oleh akun AWS *111122223333*. Sebelum menggunakan perintah ini, ganti *user input placeholder* nilai-nilai ini dan Wilayah AWS kode dengan nilai yang sesuai untuk kasus penggunaan Anda.

```
aws s3api put-object --bucket arn:aws:s3-outposts:us-
east-1:111122223333:outpost/op-01ac5d28a6a232904/accesspoint/example-outpost-access-
point \
--body my_image.jpg --key images/my_image.jpg
```

Untuk informasi selengkapnya tentang perintah ini, lihat put-object dalam Referensi AWS CLI .

> **Note**
>
> Jika operasi ini menghasilkan kesalahan Sumber Daya tidak ditemukan atau tidak responsif, VPC Anda mungkin tidak memiliki titik akhir bersama.
> Untuk memeriksa apakah ada titik akhir bersama, gunakan list-shared-endpoints AWS CLI perintah. Jika tidak ada titik akhir bersama, bekerja dengan pemilik Outpost untuk membuatnya. Untuk informasi selengkapnya, lihat ListSharedEndpointsdi Referensi API Amazon Simple Storage Service.

Example : Membuat titik akhir

Berikut adalah contoh membuat titik akhir di Outpost bersama. Sebelum menggunakan perintah ini, ganti *`user input placeholder`* nilai untuk ID Outpost, ID subnet, dan ID grup keamanan dengan nilai yang sesuai untuk kasus penggunaan Anda.

> **Note**
>
> Pengguna dapat melakukan operasi ini hanya jika pembagian sumber daya menyertakan sumber daya Outposts.

```
aws s3outposts create-endpoint --outposts-id op-01ac5d28a6a232904 --subnet-id XXXXXX --
security-group-id XXXXXXX
```

Untuk informasi selengkapnya tentang perintah ini, lihat buat-titik akhir di Referensi.AWS CLI

LainnyaLayanan AWSS3 di Outposts

LainnyaLayanan AWSyang menjalankan lokal untuk AndaAWS Outpostsjuga dapat menggunakan kapasitas Amazon S3 di Outposts Anda. Di AmazonCloudWatchsang`S3Outposts`namespace menunjukkan metrik terperinci untuk ember dalam S3 pada Outposts, tetapi metrik ini tidak termasuk

penggunaan untuk lainnyaLayanan AWS. Mengelola S3 di kapasitas Outposts Anda yang dikonsumsi oleh lainnyaLayanan AWS, lihat informasi di tabel berikut.

Layanan AWS	Deskripsi	Pelajari selengkapnya
Amazon S3	Semua S3 langsung pada penggunaan Outposts memiliki akun yang cocok dan bucketCloudWatchmetrik.	Lihat metrik
Amazon Elastic Block Store (Amazon EBS)	Untuk Amazon EBS di Outposts, Anda dapat memilihAWSOutpost sebagai tujuan snapshot Anda dan menyimpan secara lokal di S3 Anda di Outpost.	Pelajari selengkapnya
Amazon Relational Database Service (Amazon RDS)	Anda dapat menggunakan backup lokal Amazon RDS untuk menyimpan cadangan RDS Anda secara lokal di Outpost Anda.	Pelajari selengkapnya

Memantau S3 di Outposts

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di AWS Outposts dan dengan mudah menyimpan dan mengambil objek di tempat untuk aplikasi yang memerlukan akses data lokal, pemrosesan data lokal, dan residensi data. S3 on Outposts menyediakan kelas penyimpanan baru, S3 Outposts `OUTPOSTS` (), yang menggunakan API Amazon S3, dan dirancang untuk menyimpan data secara tahan lama dan berlebihan di beberapa perangkat dan server di Anda. AWS Outposts Anda berkomunikasi dengan bucket Outposts menggunakan titik akses dan koneksi titik akhir melalui cloud privat virtual (VPC). Anda dapat menggunakan API dan fitur yang sama di bucket Outposts seperti yang Anda lakukan di bucket Amazon S3, termasuk kebijakan akses, enkripsi, dan pemberian tag. Anda dapat menggunakan S3 di Outposts melalui AWS Management Console AWS Command Line Interface ,AWS CLI(), SDK AWS , atau REST API. Untuk informasi selengkapnya, lihat Apa itu Amazon S3 di Outposts?

Untuk informasi selengkapnya tentang memantau Amazon S3 di kapasitas penyimpanan Outposts, lihat topik-topik berikut.

Topik

- Mengelola kapasitas S3 pada Outposts dengan metrik Amazon CloudWatch
- Menerima pemberitahuan acara S3 di Outposts dengan menggunakan Amazon Events CloudWatch
- Memantau S3 di AWS CloudTrail Outposts dengan log

Mengelola kapasitas S3 pada Outposts dengan metrik Amazon CloudWatch

Untuk membantu mengelola kapasitas S3 tetap di Outpost Anda, kami sarankan Anda membuat CloudWatch peringatan yang memberi tahu Anda kapan pemanfaatan penyimpanan Anda melebihi ambang batas tertentu. Untuk informasi selengkapnya tentang CloudWatch metrik untuk S3 di Outposts, lihat. CloudWatch metrik Jika tidak tersedia cukup ruang untuk menyimpan objek di Outpost Anda, API mengembalikan pengecualian kapasitas yang tidak memadai (ICE). Untuk mengosongkan ruang, Anda dapat membuat CloudWatch alarm yang memicu penghapusan data eksplisit, atau menggunakan kebijakan kedaluwarsa siklus hidup untuk objek kedaluwarsa. Untuk menyimpan data sebelum dihapus, Anda dapat menggunakannya AWS DataSync untuk menyalin data dari bucket Amazon S3 di Outposts ke bucket S3 di file. Wilayah AWS Untuk informasi selengkapnya tentang penggunaan DataSync, lihat Memulai AWS DataSync di Panduan AWS DataSync Pengguna.

CloudWatch metrik

Namespace `S3Outposts` mencakup metrik berikut untuk Amazon S3 pada bucket Outposts. Anda dapat memantau jumlah total S3 di byte Outposts yang disediakan, total byte gratis yang tersedia untuk objek, dan ukuran total semua objek untuk bucket tertentu. Metrik terkait bucket atau akun ada untuk semua penggunaan S3 langsung. Penggunaan S3 tidak langsung, seperti menyimpan snapshot lokal Amazon Elastic Block Store atau backup Amazon Relational Database Service di Outpost, menghabiskan kapasitas S3, tetapi tidak disertakan dalam bucket atau metrik terkait akun. Untuk informasi selengkapnya tentang snapshot lokal Amazon EBS, lihat snapshot lokal Amazon EBS di Outposts. Untuk melihat laporan biaya Amazon EBS Anda, kunjungi https://console.aws.amazon.com/billing/.

> **Note**
>
> S3 di Outposts hanya men support metrik berikut dan bukan metrik Amazon S3 lainnya.

Karena S3 di Outposts memiliki batas kapasitas tetap, kami sarankan CloudWatch membuat alarm untuk memberi tahu Anda ketika penggunaan penyimpanan Anda melebihi ambang batas tertentu.

Metrik	Deskripsi	Periode Waktu	Unit	Jenis
`Outpost` `talByte`	Total kapasitas yang disediaka n dalam byte untuk Outposts.	5 menit	Byte	S3 di Outposts
`Outpostl` `eeBytes`	Jumlah byte gratis yang tersedia di Outposts untuk menyimpan data pelanggan.	5 menit	Byte	S3 di Outposts
`BucketU` `dBytes`	Ukuran total semua objek untuk bucket yang diberikan.	5 menit	Byte	S3 di Outposts. Penggunaan S3 langsung saja.
`Accountl` `edBytes`	Ukuran total semua objek untuk akun Outposts yang ditentukan.	5 menit	Byte	S3 di Outposts. Penggunaan S3 langsung saja.
`BytesPe` `ingRepl` `ation`	Jumlah total byte objek yang menunggu replikasi untuk aturan replikasi yang diberikan . Untuk informasi selengkap nya tentang cara mengaktifkan metrik replikasi, lihat Membuat aturan replikasi di antara Outposts.	5 menit	Byte	Tidak wajib. Untuk Replikasi S3 di Outposts.
`Operati` `sPendin` `eplicat` `n`	Jumlah total operasi yang menunggu replikasi untuk aturan replikasi yang diberikan . Untuk informasi selengkap nya tentang cara mengaktifkan metrik replikasi, lihat Membuat	5 menit	Hitungan	Tidak wajib. Untuk Replikasi S3 di Outposts.

Metrik	Deskripsi	Periode Waktu	Unit	Jenis
	aturan replikasi di antara Outposts.			
`ReplicationLatency`	Jumlah penundaan saat ini saat bucket tujuan replikasi berada di belakang bucket sumber untuk aturan replikasi yang diberikan. Untuk informasi selengkapnya tentang cara mengaktifkan metrik replikasi, lihat Membuat aturan replikasi di antara Outposts.	5 menit	Detik	Tidak wajib. Untuk Replikasi S3 di Outposts.

Menerima pemberitahuan acara S3 di Outposts dengan menggunakan Amazon Events CloudWatch

Anda dapat menggunakan CloudWatch Acara untuk membuat aturan untuk Amazon S3 pada peristiwa API Outposts. Saat membuat aturan, Anda dapat memilih untuk mendapatkan pemberitahuan melalui semua CloudWatch target yang didukung, termasuk Amazon Simple Queue Service (Amazon SQS), Amazon Simple Notification Service (Amazon SNS), dan. AWS Lambda Untuk informasi selengkapnya, lihat daftar AWS layanan yang dapat menjadi target untuk CloudWatch Acara di Panduan Pengguna CloudWatch Acara Amazon. Untuk memilih layanan target agar berfungsi dengan S3 Anda di Outposts, lihat Membuat aturan Peristiwa CloudWatch yang memicu panggilan AWS CloudTrail API AWS menggunakan Panduan Pengguna CloudWatch Amazon Events.

> **Note**
>
> Untuk S3 pada operasi objek Outposts AWS , peristiwa panggilan API yang dikirim CloudTrail oleh akan cocok dengan aturan Anda hanya jika Anda memiliki jejak (opsional dengan pemilih peristiwa) yang dikonfigurasi untuk menerima peristiwa tersebut. Untuk

informasi selengkapnya, lihat Bekerja dengan file CloudTrail log di Panduan AWS CloudTrail Pengguna.

Example

Berikut ini adalah contoh peraturan untuk operasi `DeleteObject`. Untuk menggunakan aturan contoh ini, ganti *`example-s3-bucket1`* dengan nama S3 Anda di bucket Outposts.

```
{
  "source": [
    "aws.s3-outposts"
  ],
  "detail-type": [
    "AWS API call through CloudTrail"
  ],
  "detail": {
    "eventSource": [
      "s3-outposts.amazonaws.com"
    ],
    "eventName": [
      "DeleteObject"
    ],
    "requestParameters": {
      "bucketName": [
        "example-s3-bucket1"
      ]
    }
  }
}
```

Memantau S3 di AWS CloudTrail Outposts dengan log

Amazon S3 di Outposts terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau di Layanan AWS S3 di Outposts. Anda dapat menggunakan AWS CloudTrail untuk mendapatkan informasi tentang S3 tentang permintaan tingkat bucket Outposts dan tingkat objek untuk mengaudit dan mencatat aktivitas acara S3 Anda di Outposts. Untuk mengaktifkan peristiwa CloudTrail data untuk semua bucket Outposts Anda atau untuk daftar bucket Outposts tertentu, Anda harus membuat jejak secara manual. CloudTrail Untuk informasi selengkapnya tentang entri file CloudTrail log, lihat S3 pada entri file log Outposts.

> **Note**
>
> - Merupakan praktik terbaik untuk membuat kebijakan siklus hidup untuk bucket Outposts peristiwa AWS CloudTrail data Anda. Konfigurasikan kebijakan siklus aktif untuk menghapus berkas log secara berkala setelah periode saat Anda perlu mengauditnya. Hal ini mengurangi jumlah data yang dianalisis Amazon Athena untuk setiap kueri. Untuk informasi selengkapnya, lihat Menyetel konfigurasi siklus hidup pada bucket.
> - Untuk contoh cara menanyakan CloudTrail log, lihat posting Blog AWS Big Data Menganalisis Keamanan, Kepatuhan, dan Penggunaan Aktivitas Operasional AWS CloudTrail dan Amazon Athena.

Aktifkan CloudTrail logging untuk objek di bucket S3 pada Outposts

Anda dapat menggunakan konsol Amazon S3 untuk mengonfigurasi AWS CloudTrail jejak untuk mencatat peristiwa data untuk objek di bucket Amazon S3 di Outposts. CloudTrail mendukung logging S3 pada operasi API tingkat objek Outposts sepertiGetObject,, dan. DeleteObject PutObject Peristiwa ini disebut peristiwa data.

Secara default, CloudTrail jejak tidak mencatat peristiwa data. Namun, Anda dapat mengonfigurasi jejak untuk mencatat peristiwa data bagi bucket S3 di Outposts yang Anda tentukan, atau mencatat peristiwa data untuk semua S3 bucket Outposts di Anda. Akun AWS Untuk informasi selengkapnya, lihat Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail.

CloudTrail tidak mengisi peristiwa data dalam riwayat CloudTrail acara. Selain itu, tidak semua operasi API tingkat ember S3 di Outposts diisi dalam riwayat acara. CloudTrail Untuk informasi selengkapnya tentang cara menanyakan CloudTrail log, lihat Menggunakan pola filter CloudWatch Log Amazon dan Amazon Athena untuk menanyakan CloudTrail log di Pusat AWS Pengetahuan.

Untuk mengonfigurasi jejak guna mencatat peristiwa data bagi bucket S3 di Outposts, Anda dapat menggunakan konsol AWS CloudTrail tersebut atau konsol Amazon S3. Jika Anda mengonfigurasi jejak untuk mencatat peristiwa data untuk semua bucket S3 on Outposts di Anda Akun AWS, lebih mudah menggunakan konsol. CloudTrail Untuk informasi tentang menggunakan CloudTrail konsol untuk mengonfigurasi jejak untuk mencatat peristiwa data S3 di Outposts, lihat Peristiwa data di Panduan Pengguna AWS CloudTrail .

> **Important**
>
> Biaya tambahan berlaku untuk peristiwa data. Untuk informasi selengkapnya, lihat harga AWS CloudTrail.

Prosedur berikut menunjukkan cara menggunakan konsol Amazon S3 untuk mengonfigurasi CloudTrail jejak untuk mencatat peristiwa data untuk bucket S3 di Outposts.

> **Note**
>
> Akun AWS Yang membuat bucket memilikinya dan merupakan satu-satunya yang dapat mengonfigurasi peristiwa data S3 pada Outposts yang akan dikirim. AWS CloudTrail

Untuk mengaktifkan pencatatan peristiwa CloudTrail data untuk objek di bucket S3 di Outposts

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih bucket Outposts.
3. Pilih nama bucket Outposts yang peristiwa datanya ingin Anda log dengan menggunakan. CloudTrail
4. Pilih Properti.
5. Di bagian peristiwa AWS CloudTrail data, pilih Configure in CloudTrail.

   AWS CloudTrail Konsol terbuka.

   Anda dapat membuat CloudTrail jejak baru atau menggunakan kembali jejak yang ada dan mengonfigurasi peristiwa data S3 pada Outposts untuk dicatat di jejak Anda.
6. Pada halaman Dasbor CloudTrail konsol, pilih Buat jejak.
7. Pada halaman Langkah 1 Pilih atribut jejak, berikan nama untuk jejak, pilih bucket S3 untuk menyimpan log jejak, tentukan pengaturan lain yang Anda inginkan, lalu pilih Berikutnya.
8. Pada Langkah 2 Pilih halaman peristiwa log, di bawah Jenis acara, pilih Peristiwa data.

   Untuk jenis peristiwa Data, pilih Outposts S3. Pilih Selanjutnya.

> **Note**
>
> - Saat Anda membuat jejak dan mengonfigurasi pencatatan peristiwa data untuk S3 di Outposts, Anda harus menentukan jenis peristiwa data dengan benar.
> - Jika Anda menggunakan CloudTrail konsol, pilih Outposts S3 untuk jenis peristiwa Data. Untuk informasi tentang cara membuat jejak di CloudTrail konsol, lihat Membuat dan memperbarui jejak dengan konsol di Panduan AWS CloudTrail Pengguna. Untuk informasi tentang cara mengonfigurasi log peristiwa data S3 di Outposts di CloudTrail konsol, lihat Mencatat peristiwa data untuk Objek Amazon S3 di Panduan Pengguna.AWS CloudTrail
> - Jika Anda menggunakan AWS Command Line Interface (AWS CLI) atau AWS SDK, setel `resources.type` bidang ke`AWS::S3Outposts::Object`. Untuk informasi selengkapnya tentang cara mencatat peristiwa data S3 di Outposts dengan AWS CLI, lihat Log S3 pada peristiwa Outposts di Panduan Pengguna.AWS CloudTrail
> - Jika Anda menggunakan CloudTrail konsol atau konsol Amazon S3 untuk mengonfigurasi jejak untuk mencatat peristiwa data untuk bucket S3 di Outposts, konsol Amazon S3 menunjukkan bahwa pencatatan tingkat objek diaktifkan untuk bucket.

9. Pada halaman Langkah 3 Tinjau dan buat, tinjau atribut jejak dan peristiwa log yang Anda konfigurasikan. Kemudian, pilih Buat jejak.

Untuk menonaktifkan pencatatan peristiwa CloudTrail data untuk objek di bucket S3 di Outposts

1. Masuk ke AWS Management Console dan buka CloudTrail konsol di https://console.aws.amazon.com/cloudtrail/.
2. Di panel navigasi kiri, pilih Jalur.
3. Pilih nama jejak yang Anda buat untuk mencatat peristiwa untuk bucket S3 on Outposts Anda.
4. Di halaman detail untuk jejak Anda, pilih Hentikan pencatatan di sudut kanan atas.
5. Pada kotak dialog yang muncul, pilih Hentikan pencatatan.

Mengembangkan dengan Amazon S3 di Outposts

Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 di Outposts AWS Anda serta dengan mudah menyimpan dan mengambil objek on-premise untuk aplikasi yang memerlukan akses data lokal, pengolahan data lokal, dan residensi data. S3 di Outposts menyediakan kelas penyimpanan baru, S3 Outposts (`OUTPOSTS`), yang menggunakan API Amazon S3, serta dirancang untuk menyimpan data secara tahan lama dan berlebihan di beberapa perangkat dan server di AWS Outposts Anda. Anda berkomunikasi dengan bucket Outposts menggunakan titik akses dan koneksi titik akhir melalui cloud privat virtual (VPC). Anda dapat menggunakan API dan fitur yang sama di bucket Outposts seperti yang Anda lakukan di bucket Amazon S3, termasuk kebijakan akses, enkripsi, dan pemberian tag. Anda dapat menggunakan S3 di Outposts melalui AWS Management Console, AWS Command Line Interface (AWS CLI), SDK AWS, atau API REST. Lihat informasi yang lebih lengkap di Apa itu Amazon S3 di Outposts?

Topik berikut menyediakan informasi tentang pengembangan dengan S3 di Outposts.

Topik

- Amazon S3 di operasi API Outposts
- Mengonfigurasi klien kontrol S3 di Outposts dengan menggunakan SDK for Java
- Membuat permintaan ke S3 di Outposts melalui IPv6

Amazon S3 di operasi API Outposts

Topik ini mencantumkan operasi Amazon S3, Amazon S3 Control, dan Amazon S3 on Outposts API yang dapat Anda gunakan dengan Amazon S3 di Outposts.

Topik

- Operasi API Amazon S3 untuk mengelola objek
- Operasi Amazon S3 Control API untuk mengelola bucket
- S3 di operasi API Outposts untuk mengelola Outposts

Operasi API Amazon S3 untuk mengelola objek

S3 di Outposts dirancang untuk menggunakan operasi API objek yang sama dengan Amazon S3. Anda harus menggunakan titik akses untuk mengakses objek apapun dalam bucket Outposts. Saat Anda menggunakan operasi API objek dengan S3 di Outposts, Anda memberikan titik akses

Outposts Amazon Resource Name (ARN) atau alias titik akses. Untuk informasi selengkapnya tentang alias titik akses, lihatMenggunakan alias untuk titik akses bucket S3 di Outposts Anda di bucket S3 di Outposts.

Amazon S3 di Outposts mendukung operasi API Amazon S3 berikut:

- AbortMultipartUpload
- CompleteMultipartUpload
- CopyObject
- CreateMultipartUpload
- DeleteObject
- DeleteObjects
- DeleteObjectTagging
- GetObject
- GetObjectTagging
- HeadBucket
- HeadObject
- ListMultipartUploads
- ListObjects
- ListObjectsV2
- ListObjectVersions
- ListParts
- PutObject
- PutObjectTagging
- UploadPart
- UploadPartCopy

Operasi Amazon S3 Control API untuk mengelola bucket

S3 di Outposts mendukung operasi Amazon S3 Control API berikut untuk bekerja dengan bucket.

- CreateAccessPoint
- CreateBucket

- DeleteAccessPoint
- DeleteAccessPointPolicy
- DeleteBucket
- DeleteBucketLifecycleConfiguration
- DeleteBucketPolicy
- DeleteBucketReplication
- DeleteBucketTagging
- GetAccessPoint
- GetAccessPointPolicy
- GetBucket
- GetBucketLifecycleConfiguration
- GetBucketPolicy
- GetBucketReplication
- GetBucketTagging
- GetBucketVersioning
- ListAccessPoints
- ListRegionalBuckets
- PutAccessPointPolicy
- PutBucketLifecycleConfiguration
- PutBucketPolicy
- PutBucketReplication
- PutBucketTagging
- PutBucketVersioning

S3 di operasi API Outposts untuk mengelola Outposts

S3 di Outposts mendukung operasi Amazon S3 di Outposts mendukung operasi API berikut untuk mengelola titik akhir.

- CreateEndpoint
- DeleteEndpoint
- ListEndpoints

- ListOutpostsWithS3
- ListSharedEndpoints

Mengonfigurasi klien kontrol S3 di Outposts dengan menggunakan SDK for Java

Contoh berikut mengonfigurasi klien kontrol Amazon S3 di Outposts dengan menggunakan klien kontrol Amazon S3 di Outposts dengan menggunakanAWS SDK for Java. Untuk menggunakan contoh ini, ganti masing-masing*user input placeholder*dengan informasi Anda sendiri.

```
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.s3control.AWSS3Control;
import com.amazonaws.services.s3control.AWSS3ControlClient;

public AWSS3Control createS3ControlClient() {

    String accessKey = AWSAccessKey;
    String secretKey = SecretAccessKey;
    BasicAWSCredentials awsCreds = new BasicAWSCredentials(accessKey, secretKey);

    return AWSS3ControlClient.builder().enableUseArnRegion()
            .withCredentials(new AWSStaticCredentialsProvider(awsCreds))
            .build();

}
```

Membuat permintaan ke S3 di Outposts melalui IPv6

Amazon S3 di Outposts dan S3 di Outposts dual-stack endpoint mendukung permintaan ke S3 on Outposts bucket menggunakan protokol IPv6 atau IPv4. Dengan dukungan IPv6 untuk S3 di Outposts, Anda dapat mengakses dan mengoperasikan bucket dan mengontrol sumber daya pesawat melalui S3 on Outposts API melalui jaringan IPv6.

> **Note**
>
> Tindakan objek S3 pada Outposts (`PutObject`seperti `GetObject` atau) tidak didukung melalui jaringan IPv6.

Tidak ada biaya tambahan untuk mengakses S3 di Outposts melalui jaringan IPv6. Untuk informasi lebih lanjut tentang S3 di Outposts, lihat harga S3 di Outposts.

Topik

- Memulai dengan IPv6
- Menggunakan titik akhir dual-stack untuk membuat permintaan melalui jaringan IPv6
- Menggunakan alamat IPv6 di kebijakan IAM
- Menguji kompatibilitas alamat IP
- Menggunakan IPv6 dengan AWS PrivateLink
- Menggunakan S3 pada titik akhir dual-stack Outposts

Memulai dengan IPv6

Untuk membuat permintaan ke bucket S3 on Outposts melalui IPv6, Anda harus menggunakan endpoint dual-stack. Bagian selanjutnya menjelaskan cara untuk membuat permintaan melalui IPv6 dengan menggunakan titik akhir tumpukan ganda.

Berikut ini adalah pertimbangan penting sebelum mencoba mengakses bucket S3 on Outposts melalui IPv6:

- Klien dan jaringan yang mengakses bucket harus diaktifkan agar dapat menggunakan IPv6.
- Permintaan cara hosting virtual dan cara jejak didukung untuk akses IPv6. Untuk informasi selengkapnya, lihat Menggunakan S3 pada titik akhir dual-stack Outposts.
- Jika Anda menggunakan pemfilteran alamat IP sumber di pengguna AWS Identity and Access Management (IAM) atau kebijakan bucket S3 on Outposts, Anda harus memperbarui kebijakan untuk menyertakan rentang alamat IPv6.

> **Note**
>
> Persyaratan ini hanya berlaku untuk operasi bucket S3 pada Outposts dan sumber daya bidang kontrol di seluruh jaringan IPv6. Tindakan objek Amazon S3 di Outposts tidak didukung di seluruh jaringan IPv6.

- Saat menggunakan IPv6, berkas log akses server mengeluarkan alamat IP dalam format IPv6. Anda harus memperbarui alat, skrip, dan perangkat lunak yang ada yang Anda gunakan untuk mengurai S3 pada file log Outposts, sehingga mereka dapat mengurai alamat IP jarak jauh

berformat IPv6. Alat, skrip, dan perangkat lunak yang diperbarui kemudian akan mengurai alamat IP jarak jauh yang diformat IPv6 dengan benar.

Menggunakan titik akhir dual-stack untuk membuat permintaan melalui jaringan IPv6

Untuk membuat permintaan dengan S3 pada panggilan Outposts API melalui IPv6, Anda dapat menggunakan titik akhir dual-stack melalui atau SDK. AWS CLI AWS Operasi API kontrol Amazon S3 dan operasi S3 pada Outposts API bekerja dengan cara yang sama apakah Anda mengakses S3 di Outposts melalui protokol IPv6 atau protokol IPv4. Namun, ketahuilah bahwa tindakan objek S3 pada Outposts (`PutObject`seperti `GetObject` atau) tidak didukung melalui jaringan IPv6.

Saat menggunakan AWS Command Line Interface (AWS CLI) dan AWS SDK, Anda dapat menggunakan parameter atau tanda bendera untuk mengubah ke titik akhir tumpukan ganda. Anda juga dapat menentukan titik akhir dual-stack secara langsung sebagai penggantian titik akhir S3 on Outposts dalam file konfigurasi.

Anda dapat menggunakan titik akhir dual-stack untuk mengakses bucket S3 di Outposts melalui IPv6 dari salah satu dari berikut ini:

- AWS CLI, lihat Menggunakan titik akhir tumpukan ganda dari AWS CLI.
- AWS SDK, lihat Menggunakan S3 di Outposts dual-stack endpoint dari SDK AWS.

Menggunakan alamat IPv6 di kebijakan IAM

Sebelum mencoba mengakses bucket S3 di Outposts menggunakan protokol IPv6, pastikan bahwa pengguna IAM atau kebijakan bucket S3 on Outposts yang digunakan untuk pemfilteran alamat IP diperbarui untuk menyertakan rentang alamat IPv6. Jika kebijakan pemfilteran alamat IP tidak diperbarui untuk menangani alamat IPv6, Anda dapat kehilangan akses ke bucket S3 on Outposts saat mencoba menggunakan protokol IPv6.

Kebijakan IAM yang memfilter alamat IP menggunakan operator kondisi alamat IP. Kebijakan bucket S3 on Outposts berikut mengidentifikasi rentang IP 54.240.143.* dari alamat IPv4 yang diizinkan dengan menggunakan operator kondisi alamat IP. Alamat IP apa pun di luar rentang ini akan ditolak aksesnya ke bucket `DOC-EXAMPLE-BUCKET` S3 on Outposts (). Karena semua alamat IPv6 di luar rentang yang diizinkan, kebijakan ini mencegah alamat IPv6 agar tidak dapat mengakses `DOC-EXAMPLE-BUCKET`.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3outposts:*",
      "Resource": "arn:aws:s3-outposts:region:111122223333:outpost/OUTPOSTS-ID/
bucket/DOC-EXAMPLE-BUCKET/*",
      "Condition": {
         "IpAddress": {"aws:SourceIp": "54.240.143.0/24"}
      }
    }
  ]
}
```

Anda dapat memodifikasi elemen kebijakan `Condition` bucket S3 pada Outposts untuk mengizinkan rentang alamat IPv4 `54.240.143.0/24` () dan IPv6 `2001:DB8:1234:5678::/64` () seperti yang ditunjukkan pada contoh berikut. Anda dapat menggunakan tipe blok `Condition` yang ditampilkan dalam contoh untuk memperbarui kebijakan pengguna IAM dan bucket Anda.

```
       "Condition": {
         "IpAddress": {
            "aws:SourceIp": [
              "54.240.143.0/24",
               "2001:DB8:1234:5678::/64"
             ]
          }
        }
```

Sebelum menggunakan IPv6 Anda harus memperbarui semua kebijakan pengguna IAM dan bucket terkait yang menggunakan pemfilteran alamat IP untuk memungkinkan rentang alamat IPv6 . Kami menyarankan agar Anda memperbarui kebijakan IAM dengan rentang alamat IPv6 organisasi Anda selain dari rentang alamat IPv4 Anda yang telah ada. Untuk contoh kebijakan kelompok yang memungkinkan akses atas IPv6 dan IPv4, lihat Membatasi akses ke suatu Wilayah tertentu.

Anda dapat meninjau kebijakan pengguna IAM menggunakan konsol IAM di https://console.aws.amazon.com/iam/. Untuk informasi lebih lanjut tentang IAM, lihat Panduan Pengguna IAM. Untuk informasi tentang mengedit S3 pada kebijakan bucket Outposts, lihat. Menambahkan atau mengedit kebijakan bucket untuk bucket Amazon S3 di Outposts

Menguji kompatibilitas alamat IP

Jika Anda menggunakan instance Linux atau Unix, atau platform macOS X, Anda dapat menguji akses Anda ke titik akhir dual-stack melalui IPv6. Misalnya, untuk menguji koneksi ke Amazon S3 pada titik akhir Outposts melalui IPv6, gunakan perintah: `dig`

```
dig s3-outposts.us-west-2.api.aws AAAA +short
```

Jika titik akhir dual-stack Anda melalui jaringan IPv6 diatur dengan benar, `dig` perintah mengembalikan alamat IPv6 yang terhubung. Sebagai contoh:

```
dig s3-outposts.us-west-2.api.aws AAAA +short

2600:1f14:2588:4800:b3a9:1460:159f:ebce

2600:1f14:2588:4802:6df6:c1fd:ef8a:fc76

2600:1f14:2588:4801:d802:8ccf:4e04:817
```

Menggunakan IPv6 dengan AWS PrivateLink

S3 on Outposts mendukung protokol AWS PrivateLink IPv6 untuk layanan dan titik akhir. Dengan AWS PrivateLink dukungan untuk protokol IPv6, Anda dapat terhubung ke titik akhir layanan dalam VPC Anda melalui jaringan IPv6, baik dari koneksi lokal maupun pribadi lainnya. Dukungan IPv6 AWS PrivateLinkuntuk S3 di Outposts juga memungkinkan Anda untuk berintegrasi dengan titik akhir dual-stack. AWS PrivateLink Untuk langkah-langkah tentang cara mengaktifkan IPv6AWS PrivateLink, lihat Mempercepat adopsi IPv6 Anda dengan layanan dan titik akhir. AWS PrivateLink

> **Note**
>
> Untuk memperbarui jenis alamat IP yang didukung dari IPv4 ke IPv6, lihat Memodifikasi jenis alamat IP yang didukung di Panduan Pengguna. AWS PrivateLink

Menggunakan IPv6 dengan AWS PrivateLink

Jika Anda menggunakan AWS PrivateLink IPv6, Anda harus membuat titik akhir antarmuka VPC IPv6 atau dual-stack. Untuk langkah-langkah umum tentang cara membuat titik akhir VPC

menggunakanAWS Management Console, lihat Mengakses AWS layanan menggunakan titik akhir VPC antarmuka di Panduan Pengguna. AWS PrivateLink

AWS Management Console

Gunakan prosedur berikut untuk membuat titik akhir VPC antarmuka yang terhubung ke S3 di Outposts.

1. Masuk ke AWS Management Console dan buka konsol VPC di https://console.aws.amazon.com/vpc/.
2. Di panel navigasi, pilih Titik Akhir.
3. Pilih Buat Titik Akhir.
4. Untuk kategori Layanan, pilih AWSlayanan.
5. Untuk nama Layanan, pilih layanan S3 on Outposts (com.amazonaws.us-east-1.s3-outposts).
6. Untuk VPC, pilih VPC tempat Anda akan mengakses S3 di Outposts.
7. Untuk Subnet, pilih satu subnet per Availability Zone dari mana Anda akan mengakses S3 di Outposts. Anda tidak dapat memilih beberapa subnet dari Availability Zone yang sama. Untuk setiap subnet yang Anda pilih, antarmuka jaringan endpoint baru dibuat. Secara default, alamat IP dari rentang alamat IP subnet ditetapkan ke antarmuka jaringan titik akhir. Untuk menunjuk alamat IP untuk antarmuka jaringan titik akhir, pilih Tentukan alamat IP dan masukkan alamat IPv6 dari rentang alamat subnet.
8. Untuk jenis alamat IP, pilih Dualstack. Tetapkan alamat IPv4 dan IPv6 ke antarmuka jaringan titik akhir Anda. Opsi ini didukung hanya jika semua subnet yang dipilih memiliki rentang alamat IPv4 dan IPv6.
9. Untuk grup Keamanan, pilih grup keamanan untuk diasosiasikan dengan antarmuka jaringan titik akhir untuk titik akhir VPC. Secara default, grup keamanan default dikaitkan dengan VPC.
10. Untuk Kebijakan, pilih Akses penuh untuk mengizinkan semua operasi oleh semua pengguna utama pada semua sumber daya melalui titik akhir VPC. Jika tidak, pilih Kustom untuk melampirkan kebijakan titik akhir VPC yang mengontrol izin yang dimiliki kepala sekolah untuk melakukan tindakan pada sumber daya melalui titik akhir VPC. Opsi ini hanya tersedia jika layanan mendukung kebijakan titik akhir VPC. Untuk informasi selengkapnya, lihat Kebijakan Endpoint.
11. (Opsional) Untuk menambahkan tanda, pilih Tambahkan tanda baru dan masukkan kunci dan nilai tanda.
12. Pilih Buat titik akhir.

Example — Kebijakan bucket S3 pada Outposts

Untuk mengizinkan S3 di Outposts berinteraksi dengan titik akhir VPC Anda, Anda kemudian dapat memperbarui kebijakan S3 on Outposts seperti ini:

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3-outposts:*",
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

AWS CLI

> **Note**
>
> Untuk mengaktifkan jaringan IPv6 pada titik akhir VPC Anda, Anda harus `IPv6` mengatur filter untuk S3 di Outposts`SupportedIpAddressType`.

Contoh berikut menggunakan `create-vpc-endpoint` perintah untuk membuat endpoint antarmuka dual-stack baru.

```
aws ec2 create-vpc-endpoint \
--vpc-id vpc-12345678 \
--vpc-endpoint-type Interface \
--service-name com.amazonaws.us-east-1.s3-outposts \
--subnet-id subnet-12345678 \
--security-group-id sg-12345678 \
--ip-address-type dualstack \
--dns-options "DnsRecordIpType=dualstack"
```

Bergantung pada konfigurasi AWS PrivateLink layanan, koneksi titik akhir yang baru dibuat mungkin perlu diterima oleh penyedia layanan titik akhir VPC sebelum dapat digunakan. Untuk informasi selengkapnya, lihat Menerima dan menolak permintaan koneksi titik akhir di AWS PrivateLinkPanduan Pengguna.

Contoh berikut menggunakan `modify-vpc-endpoint` perintah untuk memperbarui titik akhir VPC khusus IPV-ke titik akhir dual-stack. Titik akhir dual-stack memungkinkan akses ke jaringan IPv4 dan IPv6.

```
aws ec2 modify-vpc-endpoint \
--vpc-endpoint-id vpce-12345678 \
--add-subnet-ids subnet-12345678 \
--remove-subnet-ids subnet-12345678 \
--ip-address-type dualstack \
--dns-options "DnsRecordIpType=dualstack"
```

Untuk informasi selengkapnya tentang cara mengaktifkan jaringan IPv6AWS PrivateLink, lihat Mempercepat adopsi IPv6 Anda dengan layanan dan titik akhir. AWS PrivateLink

Menggunakan S3 pada titik akhir dual-stack Outposts

S3 di Outposts dual-stack endpoint mendukung permintaan ke S3 pada bucket Outposts melalui IPv6 dan IPv4. Bagian ini menjelaskan cara menggunakan S3 pada titik akhir dual-stack Outposts.

Topik

- S3 pada titik akhir tumpukan ganda Outposts
- Menggunakan titik akhir tumpukan ganda dari AWS CLI
- Menggunakan S3 di Outposts dual-stack endpoint dari SDK AWS

S3 pada titik akhir tumpukan ganda Outposts

Saat Anda membuat permintaan ke titik akhir dual-stack, URL bucket S3 on Outposts akan diselesaikan ke alamat IPv6 atau IPv4. Untuk informasi selengkapnya tentang mengakses bucket S3 di Outposts melalui IPv6, lihat. Membuat permintaan ke S3 di Outposts melalui IPv6

Untuk mengakses bucket S3 di Outposts melalui endpoint dual-stack, gunakan nama endpoint gaya jalur. S3 di Outposts hanya mendukung nama titik akhir dual-stack Regional, yang berarti Anda harus menentukan Region sebagai bagian dari nama.

Untuk titik akhir FIP gaya jalur dual-stack, gunakan konvensi penamaan berikut:

```
s3-outposts-fips.region.api.aws
```

Untuk titik akhir non-FIPS dual-stack, gunakan konvensi penamaan berikut:

```
s3-outposts.region.api.aws
```

> **Note**
>
> Nama titik akhir bergaya host virtual tidak didukung di S3 di Outposts.

Menggunakan titik akhir tumpukan ganda dari AWS CLI

Bagian ini memberikan contoh perintah AWS CLI yang digunakan untuk membuat permintaan ke titik akhir tumpukan ganda. Untuk petunjuk tentang pengaturan AWS CLI, lihat Memulai dengan menggunakan AWS CLI dan SDK for Java.

Anda menetapkan nilai konfigurasi `use_dualstack_endpoint` ke `true` dalam profil dalam AWS Config file Anda untuk mengarahkan semua permintaan Amazon S3 yang dibuat oleh `s3api` AWS CLI perintah `s3` dan ke titik akhir tumpukan ganda untuk Wilayah yang ditentukan. Anda menentukan Wilayah dalam file konfigurasi atau dalam perintah menggunakan `--region` opsi.

Saat menggunakan titik akhir tumpukan ganda denganAWS CLI, hanya gaya `path` pengalamatan yang didukung. Gaya pengalamatan, yang diatur dalam file konfigurasi, menentukan apakah nama bucket ada di nama host atau di URL. Untuk informasi selengkapnya, lihat s3outposts di AWS CLI Panduan Pengguna.

Untuk menggunakan titik akhir tumpukan ganda melaluiAWS CLI, gunakan `--endpoint-url` parameter dengan `https://s3-outposts-fips.region.api.aws` titik akhir `http://s3.dualstack.region.amazonaws.com` atau untuk perintah apa pun atau. `s3control` `s3outposts`

Sebagai contoh:

```
$  aws s3control list-regional-buckets --endpoint-url https://s3-
outposts.region.api.aws
```

Menggunakan S3 di Outposts dual-stack endpoint dari SDK AWS

Bagian ini memberikan contoh cara mengakses titik akhir tumpukan ganda dengan menggunakan AWS SDK.

Contoh titik akhir tumpukan ganda AWS SDK for Java 2.x

Contoh berikut menunjukkan cara menggunakan `S3OutpostsClient` kelas `S3ControlClient` dan untuk mengaktifkan titik akhir dual-stack saat membuat klien S3 di Outposts menggunakan file. AWS SDK for Java 2.x Untuk petunjuk tentang membuat dan menguji contoh Java yang berfungsi untuk Amazon S3 di Outposts, lihat. Memulai dengan menggunakan AWS CLI dan SDK for Java

Example — Buat **`S3ControlClient`** kelas dengan titik akhir dual-stack diaktifkan

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3control.S3ControlClient;
import software.amazon.awssdk.services.s3control.model.ListRegionalBucketsRequest;
import software.amazon.awssdk.services.s3control.model.ListRegionalBucketsResponse;
import software.amazon.awssdk.services.s3control.model.S3ControlException;


public class DualStackEndpointsExample1 {

    public static void main(String[] args) {
        Region clientRegion = Region.of("us-east-1");
        String accountId = "111122223333";
        String navyId = "9876543210";

        try {
            // Create an S3ControlClient with dual-stack endpoints enabled.
            S3ControlClient s3ControlClient = S3ControlClient.builder()
                                                              .region(clientRegion)
                                                              .dualstackEnabled(true)
                                                              .build();
            ListRegionalBucketsRequest listRegionalBucketsRequest =
 ListRegionalBucketsRequest.builder()

       .accountId(accountId)

       .outpostId(navyId)

       .build();

            ListRegionalBucketsResponse listBuckets =
 s3ControlClient.listRegionalBuckets(listRegionalBucketsRequest);
```

```
            System.out.printf("ListRegionalBuckets Response: %s%n",
 listBuckets.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 on Outposts
 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        }
        catch (S3ControlException e) {
            // Unknown exceptions will be thrown as an instance of this type.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 on Outposts couldn't be contacted for a response, or the
 client
            // couldn't parse the response from Amazon S3 on Outposts.
            e.printStackTrace();
        }
    }
}
```

Example — Buat **`S3OutpostsClient`** dengan titik akhir dual-stack diaktifkan

```
import com.amazonaws.AmazonServiceException;
import com.amazonaws.SdkClientException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3outposts.S3OutpostsClient;
import software.amazon.awssdk.services.s3outposts.model.ListEndpointsRequest;
import software.amazon.awssdk.services.s3outposts.model.ListEndpointsResponse;
import software.amazon.awssdk.services.s3outposts.model.S3OutpostsException;


public class DualStackEndpointsExample2 {

    public static void main(String[] args) {
        Region clientRegion = Region.of("us-east-1");

        try {
            // Create an S3OutpostsClient with dual-stack endpoints enabled.
            S3OutpostsClient s3OutpostsClient = S3OutpostsClient.builder()
                                                                .region(clientRegion)
                                                                .dualstackEnabled(true)
                                                                .build();
```

```
            ListEndpointsRequest listEndpointsRequest =
 ListEndpointsRequest.builder().build();

            ListEndpointsResponse listEndpoints =
 s3OutpostsClient.listEndpoints(listEndpointsRequest);
            System.out.printf("ListEndpoints Response: %s%n",
 listEndpoints.toString());
        } catch (AmazonServiceException e) {
            // The call was transmitted successfully, but Amazon S3 on Outposts
 couldn't process
            // it, so it returned an error response.
            e.printStackTrace();
        }
        catch (S3OutpostsException e) {
            // Unknown exceptions will be thrown as an instance of this type.
            e.printStackTrace();
        } catch (SdkClientException e) {
            // Amazon S3 on Outposts couldn't be contacted for a response, or the
 client
            // couldn't parse the response from Amazon S3 on Outposts.
            e.printStackTrace();
        }
    }
}
```

Jika Anda menggunakan AWS SDK for Java 2.x pada Windows, Anda mungkin harus mengatur properti Java virtual machine (JVM) berikut:

```
java.net.preferIPv6Addresses=true
```

Contoh kode untuk Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara menggunakan Amazon S3 dengan kit pengembangan AWS perangkat lunak (SDK).

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Meskipun tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks pada skenario terkait dan contoh lintas layanan.

Skenario adalah contoh kode yang menunjukkan cara menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan yang sama.

Contoh lintas layanan adalah contoh aplikasi yang bekerja di beberapa Layanan AWS.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai

Halo Amazon S3

Contoh kode berikut ini menunjukkan cara memulai menggunakan Amazon S3.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Kode untuk file CMake MakeLists C.txt.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS s3)
```

```
# Set this project's name.
project("hello_s3")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
    # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.

    # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
    # and set the proper subdirectory to the executables' location.

    AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_s3.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Kode untuk file sumber hello_s3.cpp.

```
#include <aws/core/Aws.h>
```

```
#include <aws/s3/S3Client.h>
#include <iostream>
#include <aws/core/auth/AWSCredentialsProviderChain.h>
using namespace Aws;
using namespace Aws::Auth;

/*
 *  A "Hello S3" starter application which initializes an Amazon Simple Storage
 Service (Amazon S3) client
 *  and lists the Amazon S3 buckets in the selected region.
 *
 *  main function
 *
 *  Usage: 'hello_s3'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        // You don't normally have to test that you are authenticated. But the
 S3 service permits anonymous requests, thus the s3Client will return "success"
 and 0 buckets even if you are unauthenticated, which can be confusing to a new
 user.
        auto provider =
 Aws::MakeShared<DefaultAWSCredentialsProviderChain>("alloc-tag");
        auto creds = provider->GetAWSCredentials();
        if (creds.IsEmpty()) {
            std::cerr << "Failed authentication" << std::endl;
        }

        Aws::S3::S3Client s3Client(clientConfig);
        auto outcome = s3Client.ListBuckets();

        if (!outcome.IsSuccess()) {
```

```
            std::cerr << "Failed with error: " << outcome.GetError() <<
 std::endl;
            result = 1;
        } else {
            std::cout << "Found " << outcome.GetResult().GetBuckets().size()
                      << " buckets\n";
            for (auto &bucket: outcome.GetResult().GetBuckets()) {
                std::cout << bucket.GetName() << std::endl;
            }
        }
    }

    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Untuk detail API, lihat ListBucketsdi Referensi AWS SDK for C++ API.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
package main

import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/s3"
)

// main uses the AWS SDK for Go V2 to create an Amazon Simple Storage Service
// (Amazon S3) client and list up to 10 buckets in your account.
```

```
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 s3Client := s3.NewFromConfig(sdkConfig)
 count := 10
 fmt.Printf("Let's list up to %v buckets for your account.\n", count)
 result, err := s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
 if err != nil {
  fmt.Printf("Couldn't list buckets for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Buckets) == 0 {
  fmt.Println("You don't have any buckets!")
 } else {
  if count > len(result.Buckets) {
   count = len(result.Buckets)
  }
  for _, bucket := range result.Buckets[:count] {
   fmt.Printf("\t%v\n", *bucket.Name)
  }
 }
}
```

- Untuk detail API, lihat ListBucketsdi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.Bucket;
import software.amazon.awssdk.services.s3.model.ListBucketsResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class HelloS3 {
    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        listBuckets(s3);
    }

    public static void listBuckets(S3Client s3) {
        try {
            ListBucketsResponse response = s3.listBuckets();
            List<Bucket> bucketList = response.buckets();
            bucketList.forEach(bucket -> {
                System.out.println("Bucket Name: " + bucket.name());
            });

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat ListBucketsdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";

// When no region or credentials are provided, the SDK will use the
// region and credentials from the local AWS config.
const client = new S3Client({});

export const helloS3 = async () => {
  const command = new ListBucketsCommand({});

  const { Buckets } = await client.send(command);
  console.log("Buckets: ");
  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
  return Buckets;
};
```

- Untuk detail API, lihat ListBucketsdi Referensi AWS SDK for JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
use Aws\S3\S3Client;
```

```
$client = new S3Client(['region' => 'us-west-2']);
$results = $client->listBuckets();
var_dump($results);
```

- Untuk detail API, lihat ListBucketsdi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import boto3


def hello_s3():
    """
    Use the AWS SDK for Python (Boto3) to create an Amazon Simple Storage Service
    (Amazon S3) resource and list the buckets in your account.
    This example uses the default settings specified in your shared credentials
    and config files.
    """
    s3_resource = boto3.resource("s3")
    print("Hello, Amazon S3! Let's list your buckets:")
    for bucket in s3_resource.buckets.all():
        print(f"\t{bucket.name}")


if __name__ == "__main__":
    hello_s3()
```

- Untuk detail API, lihat ListBucketsdi AWS SDK for Python (Boto3) Referensi API.

Contoh kode

- Tindakan untuk Amazon S3 menggunakan SDK AWS
  - Gunakan AbortMultipartUpload dengan AWS SDK atau CLI
  - Gunakan AbortMultipartUploads dengan AWS SDK atau CLI
  - Gunakan CompleteMultipartUpload dengan AWS SDK atau CLI
  - Gunakan CopyObject dengan AWS SDK atau CLI
  - Gunakan CreateBucket dengan AWS SDK atau CLI
  - Gunakan CreateMultiRegionAccessPoint dengan AWS SDK atau CLI
  - Gunakan CreateMultipartUpload dengan AWS SDK atau CLI
  - Gunakan DeleteBucket dengan AWS SDK atau CLI
  - Gunakan DeleteBucketAnalyticsConfiguration dengan AWS SDK atau CLI
  - Gunakan DeleteBucketCors dengan AWS SDK atau CLI
  - Gunakan DeleteBucketEncryption dengan AWS SDK atau CLI
  - Gunakan DeleteBucketInventoryConfiguration dengan AWS SDK atau CLI
  - Gunakan DeleteBucketLifecycle dengan AWS SDK atau CLI
  - Gunakan DeleteBucketMetricsConfiguration dengan AWS SDK atau CLI
  - Gunakan DeleteBucketPolicy dengan AWS SDK atau CLI
  - Gunakan DeleteBucketReplication dengan AWS SDK atau CLI
  - Gunakan DeleteBucketTagging dengan AWS SDK atau CLI
  - Gunakan DeleteBucketWebsite dengan AWS SDK atau CLI
  - Gunakan DeleteObject dengan AWS SDK atau CLI
  - Gunakan DeleteObjectTagging dengan AWS SDK atau CLI
  - Gunakan DeleteObjects dengan AWS SDK atau CLI
  - Gunakan DeletePublicAccessBlock dengan AWS SDK atau CLI
  - Gunakan GetBucketAccelerateConfiguration dengan AWS SDK atau CLI
  - Gunakan GetBucketAcl dengan AWS SDK atau CLI
  - Gunakan GetBucketAnalyticsConfiguration dengan AWS SDK atau CLI
  - Gunakan GetBucketCors dengan AWS SDK atau CLI
  - Gunakan GetBucketEncryption dengan AWS SDK atau CLI
  - Gunakan GetBucketInventoryConfiguration dengan AWS SDK atau CLI

  - Gunakan GetBucketLifecycleConfiguration dengan AWS SDK atau CLI

- Gunakan GetBucketLocation dengan AWS SDK atau CLI
- Gunakan GetBucketLogging dengan AWS SDK atau CLI
- Gunakan GetBucketMetricsConfiguration dengan AWS SDK atau CLI
- Gunakan GetBucketNotification dengan AWS SDK atau CLI
- Gunakan GetBucketPolicy dengan AWS SDK atau CLI
- Gunakan GetBucketPolicyStatus dengan AWS SDK atau CLI
- Gunakan GetBucketReplication dengan AWS SDK atau CLI
- Gunakan GetBucketRequestPayment dengan AWS SDK atau CLI
- Gunakan GetBucketTagging dengan AWS SDK atau CLI
- Gunakan GetBucketVersioning dengan AWS SDK atau CLI
- Gunakan GetBucketWebsite dengan AWS SDK atau CLI
- Gunakan GetObject dengan AWS SDK atau CLI
- Gunakan GetObjectAcl dengan AWS SDK atau CLI
- Gunakan GetObjectLegalHold dengan AWS SDK atau CLI
- Gunakan GetObjectLockConfiguration dengan AWS SDK atau CLI
- Gunakan GetObjectRetention dengan AWS SDK atau CLI
- Gunakan GetObjectTagging dengan AWS SDK atau CLI
- Gunakan GetPublicAccessBlock dengan AWS SDK atau CLI
- Gunakan HeadBucket dengan AWS SDK atau CLI
- Gunakan HeadObject dengan AWS SDK atau CLI
- Gunakan ListBucketAnalyticsConfigurations dengan AWS SDK atau CLI
- Gunakan ListBucketInventoryConfigurations dengan AWS SDK atau CLI
- Gunakan ListBuckets dengan AWS SDK atau CLI
- Gunakan ListMultipartUploads dengan AWS SDK atau CLI
- Gunakan ListObjectVersions dengan AWS SDK atau CLI
- Gunakan ListObjects dengan AWS SDK atau CLI
- Gunakan ListObjectsV2 dengan AWS SDK atau CLI
- Gunakan PutBucketAccelerateConfiguration dengan AWS SDK atau CLI
- Gunakan PutBucketAcl dengan AWS SDK atau CLI

- Gunakan PutBucketCors dengan AWS SDK atau CLI

- Gunakan PutBucketEncryption dengan AWS SDK atau CLI
- Gunakan PutBucketLifecycleConfiguration dengan AWS SDK atau CLI
- Gunakan PutBucketLogging dengan AWS SDK atau CLI
- Gunakan PutBucketNotification dengan AWS SDK atau CLI
- Gunakan PutBucketNotificationConfiguration dengan AWS SDK atau CLI
- Gunakan PutBucketPolicy dengan AWS SDK atau CLI
- Gunakan PutBucketReplication dengan AWS SDK atau CLI
- Gunakan PutBucketRequestPayment dengan AWS SDK atau CLI
- Gunakan PutBucketTagging dengan AWS SDK atau CLI
- Gunakan PutBucketVersioning dengan AWS SDK atau CLI
- Gunakan PutBucketWebsite dengan AWS SDK atau CLI
- Gunakan PutObject dengan AWS SDK atau CLI
- Gunakan PutObjectAcl dengan AWS SDK atau CLI
- Gunakan PutObjectLegalHold dengan AWS SDK atau CLI
- Gunakan PutObjectLockConfiguration dengan AWS SDK atau CLI
- Gunakan PutObjectRetention dengan AWS SDK atau CLI
- Gunakan RestoreObject dengan AWS SDK atau CLI
- Gunakan SelectObjectContent dengan AWS SDK atau CLI
- Gunakan UploadPart dengan AWS SDK atau CLI

- Skenario untuk Amazon S3 menggunakan SDK AWS
  - Membuat URL yang telah ditetapkan sebelumnya untuk Amazon S3 menggunakan SDK AWS
  - Halaman web yang mencantumkan objek Amazon S3 menggunakan SDK AWS
  - Hapus unggahan multipart yang tidak lengkap ke Amazon S3 menggunakan SDK AWS
  - Mengunduh semua objek di bucket Amazon Simple Storage Service (Amazon S3) ke direktori lokal
  - Mendapatkan objek Amazon S3 dari Titik Akses Multi-Wilayah dengan menggunakan SDK AWS
  - Dapatkan objek dari bucket Amazon S3 menggunakan AWS SDK, dengan menentukan header If-Modified-Since
  - Memulai bucket dan objek Amazon S3 menggunakan SDK AWS

  - Memulai enkripsi untuk objek Amazon S3 menggunakan SDK AWS
  - Memulai tag untuk objek Amazon S3 menggunakan SDK AWS

- Dapatkan konfigurasi penahanan hukum objek Amazon S3 menggunakan SDK AWS
- Bekerja dengan fitur kunci objek Amazon S3 menggunakan SDK AWS
- Mengelola daftar kontrol akses (ACL) untuk bucket Amazon S3 menggunakan SDK AWS
- Mengelola objek Amazon S3 berversi dalam batch dengan fungsi Lambda menggunakan SDK AWS
- Mengurai URI Amazon S3 menggunakan SDK AWS
- Melakukan salinan multi-bagian objek Amazon S3 menggunakan SDK AWS
- Melakukan pengunggahan multi-bagian objek Amazon S3 menggunakan SDK AWS
- Lacak unggahan atau unduh objek Amazon S3 menggunakan SDK AWS
- Contoh pendekatan untuk pengujian unit dan integrasi dengan AWS SDK
- Mengunggah direktori lokal secara rekursif ke bucket Amazon Simple Storage Service (Amazon S3)
- Unggah atau unduh file besar ke dan dari Amazon S3 menggunakan SDK AWS
- Unggah aliran dengan ukuran yang tidak diketahui ke objek Amazon S3 menggunakan SDK AWS
- Menggunakan checksum untuk bekerja dengan objek Amazon S3 menggunakan SDK AWS
- Bekerja dengan objek berversi Amazon S3 menggunakan SDK AWS

- Contoh tanpa server untuk Amazon S3 menggunakan SDK AWS
  - Menginvokasi fungsi Lambda dari pemicu Amazon S3
- Contoh lintas layanan untuk Amazon AWS S3 menggunakan SDK
  - Membangun aplikasi Amazon Transcribe
  - Mengonversi teks menjadi ucapan dan kembali ke teks menggunakan AWS SDK
  - Membuat aplikasi manajemen aset foto yang memungkinkan pengguna mengelola foto menggunakan label
  - Membuat aplikasi penjelajah Amazon Textract
  - Mendeteksi APD dalam gambar dengan Amazon AWS Rekognition menggunakan SDK
  - Mendeteksi entitas dalam teks yang diekstrak dari gambar menggunakan SDK AWS
  - Mendeteksi wajah dalam gambar menggunakan AWS SDK
  - Mendeteksi objek dalam gambar dengan Amazon Rekognition menggunakan SDK AWS
  - Mendeteksi orang dan objek dalam video dengan Amazon Rekognition menggunakan SDK AWS

  - Simpan EXIF dan informasi gambar lainnya menggunakan SDK AWS

- Transformasi data untuk aplikasi Anda dengan S3 Object Lambda

Tindakan untuk Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara melakukan tindakan Amazon S3 individual dengan AWS SDK. Kutipan ini memanggil Amazon S3 API dan merupakan kutipan kode dari program yang lebih besar yang harus dijalankan dalam konteks. Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan instruksi untuk mengatur dan menjalankan kode.

Contoh berikut hanya mencakup tindakan yang paling umum digunakan. Untuk daftar lengkapnya, lihat Referensi API Amazon Simple Storage Service (Amazon S3).

Contoh

- Gunakan AbortMultipartUpload dengan AWS SDK atau CLI
- Gunakan AbortMultipartUploads dengan AWS SDK atau CLI
- Gunakan CompleteMultipartUpload dengan AWS SDK atau CLI
- Gunakan CopyObject dengan AWS SDK atau CLI
- Gunakan CreateBucket dengan AWS SDK atau CLI
- Gunakan CreateMultiRegionAccessPoint dengan AWS SDK atau CLI
- Gunakan CreateMultipartUpload dengan AWS SDK atau CLI
- Gunakan DeleteBucket dengan AWS SDK atau CLI
- Gunakan DeleteBucketAnalyticsConfiguration dengan AWS SDK atau CLI
- Gunakan DeleteBucketCors dengan AWS SDK atau CLI
- Gunakan DeleteBucketEncryption dengan AWS SDK atau CLI
- Gunakan DeleteBucketInventoryConfiguration dengan AWS SDK atau CLI
- Gunakan DeleteBucketLifecycle dengan AWS SDK atau CLI
- Gunakan DeleteBucketMetricsConfiguration dengan AWS SDK atau CLI
- Gunakan DeleteBucketPolicy dengan AWS SDK atau CLI
- Gunakan DeleteBucketReplication dengan AWS SDK atau CLI
- Gunakan DeleteBucketTagging dengan AWS SDK atau CLI
- Gunakan DeleteBucketWebsite dengan AWS SDK atau CLI
- Gunakan DeleteObject dengan AWS SDK atau CLI

- Gunakan DeleteObjectTagging dengan AWS SDK atau CLI
- Gunakan DeleteObjects dengan AWS SDK atau CLI
- Gunakan DeletePublicAccessBlock dengan AWS SDK atau CLI
- Gunakan GetBucketAccelerateConfiguration dengan AWS SDK atau CLI
- Gunakan GetBucketAcl dengan AWS SDK atau CLI
- Gunakan GetBucketAnalyticsConfiguration dengan AWS SDK atau CLI
- Gunakan GetBucketCors dengan AWS SDK atau CLI
- Gunakan GetBucketEncryption dengan AWS SDK atau CLI
- Gunakan GetBucketInventoryConfiguration dengan AWS SDK atau CLI
- Gunakan GetBucketLifecycleConfiguration dengan AWS SDK atau CLI
- Gunakan GetBucketLocation dengan AWS SDK atau CLI
- Gunakan GetBucketLogging dengan AWS SDK atau CLI
- Gunakan GetBucketMetricsConfiguration dengan AWS SDK atau CLI
- Gunakan GetBucketNotification dengan AWS SDK atau CLI
- Gunakan GetBucketPolicy dengan AWS SDK atau CLI
- Gunakan GetBucketPolicyStatus dengan AWS SDK atau CLI
- Gunakan GetBucketReplication dengan AWS SDK atau CLI
- Gunakan GetBucketRequestPayment dengan AWS SDK atau CLI
- Gunakan GetBucketTagging dengan AWS SDK atau CLI
- Gunakan GetBucketVersioning dengan AWS SDK atau CLI
- Gunakan GetBucketWebsite dengan AWS SDK atau CLI
- Gunakan GetObject dengan AWS SDK atau CLI
- Gunakan GetObjectAcl dengan AWS SDK atau CLI
- Gunakan GetObjectLegalHold dengan AWS SDK atau CLI
- Gunakan GetObjectLockConfiguration dengan AWS SDK atau CLI
- Gunakan GetObjectRetention dengan AWS SDK atau CLI
- Gunakan GetObjectTagging dengan AWS SDK atau CLI
- Gunakan GetPublicAccessBlock dengan AWS SDK atau CLI
- Gunakan HeadBucket dengan AWS SDK atau CLI

- Gunakan HeadObject dengan AWS SDK atau CLI
- Gunakan ListBucketAnalyticsConfigurations dengan AWS SDK atau CLI
- Gunakan ListBucketInventoryConfigurations dengan AWS SDK atau CLI
- Gunakan ListBuckets dengan AWS SDK atau CLI
- Gunakan ListMultipartUploads dengan AWS SDK atau CLI
- Gunakan ListObjectVersions dengan AWS SDK atau CLI
- Gunakan ListObjects dengan AWS SDK atau CLI
- Gunakan ListObjectsV2 dengan AWS SDK atau CLI
- Gunakan PutBucketAccelerateConfiguration dengan AWS SDK atau CLI
- Gunakan PutBucketAcl dengan AWS SDK atau CLI
- Gunakan PutBucketCors dengan AWS SDK atau CLI
- Gunakan PutBucketEncryption dengan AWS SDK atau CLI
- Gunakan PutBucketLifecycleConfiguration dengan AWS SDK atau CLI
- Gunakan PutBucketLogging dengan AWS SDK atau CLI
- Gunakan PutBucketNotification dengan AWS SDK atau CLI
- Gunakan PutBucketNotificationConfiguration dengan AWS SDK atau CLI
- Gunakan PutBucketPolicy dengan AWS SDK atau CLI
- Gunakan PutBucketReplication dengan AWS SDK atau CLI
- Gunakan PutBucketRequestPayment dengan AWS SDK atau CLI
- Gunakan PutBucketTagging dengan AWS SDK atau CLI
- Gunakan PutBucketVersioning dengan AWS SDK atau CLI
- Gunakan PutBucketWebsite dengan AWS SDK atau CLI
- Gunakan PutObject dengan AWS SDK atau CLI
- Gunakan PutObjectAcl dengan AWS SDK atau CLI
- Gunakan PutObjectLegalHold dengan AWS SDK atau CLI
- Gunakan PutObjectLockConfiguration dengan AWS SDK atau CLI
- Gunakan PutObjectRetention dengan AWS SDK atau CLI
- Gunakan RestoreObject dengan AWS SDK atau CLI
- Gunakan SelectObjectContent dengan AWS SDK atau CLI
- Gunakan UploadPart dengan AWS SDK atau CLI

Gunakan **`AbortMultipartUpload`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`AbortMultipartUpload`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Hapus unggahan multipart yang tidak lengkap

CLI

AWS CLI

Untuk membatalkan unggahan multipart yang ditentukan

`abort-multipart-upload`Perintah berikut membatalkan unggahan multibagian untuk kunci `multipart/01` di bucket. `my-bucket`

```
aws s3api abort-multipart-upload \
    --bucket my-bucket \
    --key multipart/01 \
    --upload-id
 dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3U
```

ID unggahan yang diperlukan oleh perintah ini adalah output oleh `create-multipart-upload` dan juga dapat diambil dengan`list-multipart-uploads`.

- Untuk detail API, lihat AbortMultipartUploaddi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini membatalkan unggahan multipart yang dibuat lebih awal dari 5 hari yang lalu.

```
Remove-S3MultipartUpload -BucketName test-files -DaysBefore 5
```

Contoh 2: Perintah ini membatalkan unggahan multipart yang dibuat lebih awal dari 2 Januari 2014.

```
Remove-S3MultipartUpload -BucketName test-files -InitiatedDate "Thursday, January
 02, 2014"
```

Contoh 3: Perintah ini membatalkan unggahan multipart yang dibuat lebih awal dari 2 Januari 2014, 10:45:37.

```
Remove-S3MultipartUpload -BucketName test-files -InitiatedDate "2014/01/02
 10:45:37"
```

- Untuk detail API, lihat AbortMultipartUploaddi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`AbortMultipartUploads`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`AbortMultipartUploads`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
using System;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Transfer;

/// <summary>
/// This example shows how to use the Amazon Simple Storage Service
/// (Amazon S3) to stop a multi-part upload process using the Amazon S3
/// TransferUtility.
/// </summary>
```

```
    public class AbortMPU
    {
        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";

            // If the AWS Region defined for your default user is different
            // from the Region where your Amazon S3 bucket is located,
            // pass the Region name to the S3 client object's constructor.
            // For example: RegionEndpoint.USWest2.
            IAmazonS3 client = new AmazonS3Client();

            await AbortMPUAsync(client, bucketName);
        }

        /// <summary>
        /// Cancels the multi-part copy process.
        /// </summary>
        /// <param name="client">The initialized client object used to create
        /// the TransferUtility object.</param>
        /// <param name="bucketName">The name of the S3 bucket where the
        /// multi-part copy operation is in progress.</param>
        public static async Task AbortMPUAsync(IAmazonS3 client, string
bucketName)
        {
            try
            {
                var transferUtility = new TransferUtility(client);

                // Cancel all in-progress uploads initiated before the specified
date.
                await transferUtility.AbortMultipartUploadsAsync(
                    bucketName, DateTime.Now.AddDays(-7));
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine($"Error: {e.Message}");
            }
        }
    }
```

- Untuk detail API, lihat AbortMultipartUploadsdi Referensi AWS SDK for .NET API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CompleteMultipartUpload`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CompleteMultipartUpload`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Melakukan penyalinan multibagian
- Melakukan pengunggahan multibagian
- Gunakan checksum

CLI

AWS CLI

Perintah berikut menyelesaikan unggahan multipart untuk kunci `multipart/01` di bucket: `my-bucket`

```
aws s3api complete-multipart-upload --multipart-upload file://
mpustruct --bucket my-bucket --key 'multipart/01' --upload-id
 dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3U
```

ID unggahan yang diperlukan oleh perintah ini adalah output oleh `create-multipart-upload` dan juga dapat diambil dengan`list-multipart-uploads`.

Opsi unggahan multibagian dalam perintah di atas mengambil struktur JSON yang menjelaskan bagian-bagian dari unggahan multibagian yang harus dipasang kembali ke dalam file lengkap. Dalam contoh ini, `file://` awalan digunakan untuk memuat struktur JSON dari file di folder lokal bernama. `mpustruct`

mpustruct:

```
{
  "Parts": [
    {
      "ETag": "e868e0f4719e394144ef36531ee6824c",
```

```
      "PartNumber": 1
    },
    {
      "ETag": "6bb2b12753d66fe86da4998aa33fffb0",
      "PartNumber": 2
    },
    {
      "ETag": "d0a0112e841abec9c9ec83406f0159c8",
      "PartNumber": 3
    }
  ]
}
```

Nilai ETag untuk setiap bagian yang diunggah adalah output setiap kali Anda mengunggah bagian menggunakan `upload-part` perintah dan juga dapat diambil dengan memanggil `list-parts` atau dihitung dengan mengambil checksum MD5 dari setiap bagian.

Output:

```
{
    "ETag": "\"3944a9f7a4faab7f78788ff6210f63f0-3\"",
    "Bucket": "my-bucket",
    "Location": "https://my-bucket.s3.amazonaws.com/multipart%2F01",
    "Key": "multipart/01"
}
```

- Untuk detail API, lihat CompleteMultipartUploaddi Referensi AWS CLI Perintah.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    let _complete_multipart_upload_res = client
        .complete_multipart_upload()
        .bucket(&bucket_name)
```

```
        .key(&key)
        .multipart_upload(completed_multipart_upload)
        .upload_id(upload_id)
        .send()
        .await
        .unwrap();
```

- Untuk detail API, lihat CompleteMultipartUploadreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan `CopyObject` dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CopyObject`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai bucket dan objek
- Memulai enkripsi

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
using System;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

public class CopyObject
```

```
    {
        public static async Task Main()
        {
            // Specify the AWS Region where your buckets are located if it is
            // different from the AWS Region of the default user.
            IAmazonS3 s3Client = new AmazonS3Client();

            // Remember to change these values to refer to your Amazon S3
 objects.
            string sourceBucketName = "doc-example-bucket1";
            string destinationBucketName = "doc-example-bucket2";
            string sourceObjectKey = "testfile.txt";
            string destinationObjectKey = "testfilecopy.txt";

            Console.WriteLine($"Copying {sourceObjectKey} from {sourceBucketName}
 to ");
            Console.WriteLine($"{destinationBucketName} as
 {destinationObjectKey}");

            var response = await CopyingObjectAsync(
                s3Client,
                sourceObjectKey,
                destinationObjectKey,
                sourceBucketName,
                destinationBucketName);

            if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
            {
                Console.WriteLine("\nCopy complete.");
            }
        }

        /// <summary>
        /// This method calls the AWS SDK for .NET to copy an
        /// object from one Amazon S3 bucket to another.
        /// </summary>
        /// <param name="client">The Amazon S3 client object.</param>
        /// <param name="sourceKey">The name of the object to be copied.</param>
        /// <param name="destinationKey">The name under which to save the copy.</
param>
        /// <param name="sourceBucketName">The name of the Amazon S3 bucket
        /// where the file is located now.</param>
        /// <param name="destinationBucketName">The name of the Amazon S3
        /// bucket where the copy should be saved.</param>
```

```
        /// <returns>Returns a CopyObjectResponse object with the results from
        /// the async call.</returns>
        public static async Task<CopyObjectResponse> CopyingObjectAsync(
            IAmazonS3 client,
            string sourceKey,
            string destinationKey,
            string sourceBucketName,
            string destinationBucketName)
        {
            var response = new CopyObjectResponse();
            try
            {
                var request = new CopyObjectRequest
                {
                    SourceBucket = sourceBucketName,
                    SourceKey = sourceKey,
                    DestinationBucket = destinationBucketName,
                    DestinationKey = destinationKey,
                };
                response = await client.CopyObjectAsync(request);
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error copying object: '{ex.Message}'");
            }

            return response;
        }
    }
```

- Untuk detail API, lihat CopyObjectdi Referensi AWS SDK for .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function copy_item_in_bucket
#
# This function creates a copy of the specified file in the same bucket.
#
# Parameters:
#       $1 - The name of the bucket to copy the file from and to.
#       $2 - The key of the source file to copy.
#       $3 - The key of the destination file.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function copy_item_in_bucket() {
  local bucket_name=$1
  local source_key=$2
  local destination_key=$3
  local response

  response=$(aws s3api copy-object \
    --bucket "$bucket_name" \
    --copy-source "$bucket_name/$source_key" \
    --key "$destination_key")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api copy-object operation failed.\n$response"
    return 1
  fi
}
```

- Untuk detail API, lihat CopyObjectdi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::copyObject(const Aws::String &objectKey, const Aws::String
 &fromBucket, const Aws::String &toBucket,
                            const Aws::S3::S3ClientConfiguration &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::CopyObjectRequest request;

    request.WithCopySource(fromBucket + "/" + objectKey)
            .WithKey(objectKey)
            .WithBucket(toBucket);

    Aws::S3::Model::CopyObjectOutcome outcome = client.CopyObject(request);
    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: copyObject: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;

    } else {
        std::cout << "Successfully copied " << objectKey << " from " <<
 fromBucket <<
                  " to " << toBucket << "." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat CopyObjectdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Perintah berikut menyalin objek dari `bucket-1` kebucket-2:

```
aws s3api copy-object --copy-source bucket-1/test.txt --key test.txt --bucket
 bucket-2
```

Output:

```
{
    "CopyObjectResult": {
        "LastModified": "2015-11-10T01:07:25.000Z",
        "ETag": "\"589c8b79c230a6ecd5a7e1d040a9a030\""
    },
    "VersionId": "YdnYvTCVDqRRFA.NFJjy36p0hxifMlkA"
}
```

- Untuk detail API, lihat CopyObjectdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}
```

```
// CopyToBucket copies an object in a bucket to another bucket.
func (basics BucketBasics) CopyToBucket(sourceBucket string, destinationBucket
 string, objectKey string) error {
 _, err := basics.S3Client.CopyObject(context.TODO(), &s3.CopyObjectInput{
  Bucket:     aws.String(destinationBucket),
  CopySource: aws.String(fmt.Sprintf("%v/%v", sourceBucket, objectKey)),
  Key:        aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't copy object from %v:%v to %v:%v. Here's why: %v\n",
   sourceBucket, objectKey, destinationBucket, objectKey, err)
 }
 return err
}
```

- Untuk detail API, lihat CopyObjectdi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Salin objek menggunakan S3Client.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.CopyObjectRequest;
import software.amazon.awssdk.services.s3.model.CopyObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;

/**
 * Before running this Java V2 code example, set up your development
```

```
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class CopyObject {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <objectKey> <fromBucket> <toBucket>

                Where:
                    objectKey - The name of the object (for example, book.pdf).
                    fromBucket - The S3 bucket name that contains the object (for
 example, bucket1).
                    toBucket - The S3 bucket to copy the object to (for example,
 bucket2).
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String objectKey = args[0];
        String fromBucket = args[1];
        String toBucket = args[2];
        System.out.format("Copying object %s from bucket %s to %s\n", objectKey,
 fromBucket, toBucket);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        copyBucketObject(s3, fromBucket, objectKey, toBucket);
        s3.close();
    }

    public static String copyBucketObject(S3Client s3, String fromBucket, String
 objectKey, String toBucket) {
```

```
        CopyObjectRequest copyReq = CopyObjectRequest.builder()
                .sourceBucket(fromBucket)
                .sourceKey(objectKey)
                .destinationBucket(toBucket)
                .destinationKey(objectKey)
                .build();

        try {
            CopyObjectResponse copyRes = s3.copyObject(copyReq);
            return copyRes.copyObjectResult().toString();

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

Gunakan S3 TransferManager untuk menyalin objek dari satu ember ke ember lainnya. Lihat file lengkap dan lakukan pengujian.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.services.s3.model.CopyObjectRequest;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedCopy;
import software.amazon.awssdk.transfer.s3.model.Copy;
import software.amazon.awssdk.transfer.s3.model.CopyRequest;

import java.util.UUID;

    public String copyObject(S3TransferManager transferManager, String
 bucketName,
            String key, String destinationBucket, String destinationKey) {
        CopyObjectRequest copyObjectRequest = CopyObjectRequest.builder()
                .sourceBucket(bucketName)
                .sourceKey(key)
                .destinationBucket(destinationBucket)
                .destinationKey(destinationKey)
                .build();
```

```
        CopyRequest copyRequest = CopyRequest.builder()
                .copyObjectRequest(copyObjectRequest)
                .build();

        Copy copy = transferManager.copy(copyRequest);

        CompletedCopy completedCopy = copy.completionFuture().join();
        return completedCopy.response().copyObjectResult().eTag();
    }
```

- Untuk detail API, lihat CopyObjectdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Salin objek.

```
import { S3Client, CopyObjectCommand } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new CopyObjectCommand({
    CopySource: "SOURCE_BUCKET/SOURCE_OBJECT_KEY",
    Bucket: "DESTINATION_BUCKET",
    Key: "NEW_OBJECT_KEY",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
```

```
  }
};
```

- Untuk detail API, lihat CopyObjectdi Referensi AWS SDK for JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun copyBucketObject(
    fromBucket: String,
    objectKey: String,
    toBucket: String,
) {
    var encodedUrl = ""
    try {
        encodedUrl = URLEncoder.encode("$fromBucket/$objectKey",
 StandardCharsets.UTF_8.toString())
    } catch (e: UnsupportedEncodingException) {
        println("URL could not be encoded: " + e.message)
    }

    val request =
        CopyObjectRequest {
            copySource = encodedUrl
            bucket = toBucket
            key = objectKey
        }
    S3Client { region = "us-east-1" }.use { s3 ->
        s3.copyObject(request)
    }
}
```

- Untuk detail API, lihat CopyObjectdi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Salinan sederhana dari suatu objek.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        try {
            $folder = "copied-folder";
            $this->s3client->copyObject([
                'Bucket' => $this->bucketName,
                'CopySource' => "$this->bucketName/$fileName",
                'Key' => "$folder/$fileName-copy",
            ]);
            echo "Copied $fileName to $folder/$fileName-copy.\n";
        } catch (Exception $exception) {
            echo "Failed to copy $fileName with error: " . $exception-
>getMessage();
            exit("Please fix error with object copying before continuing.");
        }
```

- Untuk detail API, lihat CopyObjectdi Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menyalin objek "sample.txt" dari bucket “test-files” ke bucket yang sama tetapi dengan kunci baru "sample-copy.txt “.

```
Copy-S3Object -BucketName test-files -Key sample.txt -DestinationKey sample-
copy.txt
```

Contoh 2: Perintah ini menyalin objek "sample.txt" dari bucket “test-files” ke bucket “backup files” dengan kunci "sample-copy.txt “.

```
Copy-S3Object -BucketName test-files -Key sample.txt -DestinationKey sample-
copy.txt -DestinationBucket backup-files
```

Contoh 3: Perintah ini mengunduh objek "sample.txt" dari bucket “test-files” ke file lokal dengan nama "local-sample.txt “.

```
Copy-S3Object -BucketName test-files -Key sample.txt -LocalFile local-sample.txt
```

Contoh 4: Mengunduh objek tunggal ke file yang ditentukan. File yang diunduh akan ditemukan di c:\downloads\data\archive.zip

```
Copy-S3Object -BucketName test-files -Key data/archive.zip -LocalFolder c:
\downloads
```

Contoh 5: Download semua objek yang cocok dengan key prefix yang ditentukan ke folder lokal. Hirarki kunci relatif akan dipertahankan sebagai subfolder di lokasi unduhan keseluruhan.

```
Copy-S3Object -BucketName test-files -KeyPrefix data -LocalFolder c:\downloads
```

- Untuk detail API, lihat CopyObjectdi Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    def copy(self, dest_object):
        """
        Copies the object to another bucket.

        :param dest_object: The destination object initialized with a bucket and
 key.
                            This is a Boto3 Object resource.
        """
        try:
            dest_object.copy_from(
                CopySource={"Bucket": self.object.bucket_name, "Key":
 self.object.key}
            )
            dest_object.wait_until_exists()
            logger.info(
                "Copied object from %s:%s to %s:%s.",
                self.object.bucket_name,
                self.object.key,
                dest_object.bucket_name,
                dest_object.key,
            )
        except ClientError:
            logger.exception(
                "Couldn't copy object from %s/%s to %s/%s.",
                self.object.bucket_name,
                self.object.key,
                dest_object.bucket_name,
                dest_object.key,
            )
            raise
```

- Untuk detail API, lihat CopyObjectdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Salin objek.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectCopyWrapper
  attr_reader :source_object

  # @param source_object [Aws::S3::Object] An existing Amazon S3 object. This is
 used as the source object for
  #                                        copy actions.
  def initialize(source_object)
    @source_object = source_object
  end

  # Copy the source object to the specified target bucket and rename it with the
 target key.
  #
  # @param target_bucket [Aws::S3::Bucket] An existing Amazon S3 bucket where the
 object is copied.
  # @param target_object_key [String] The key to give the copy of the object.
  # @return [Aws::S3::Object, nil] The copied object when successful; otherwise,
 nil.
  def copy_object(target_bucket, target_object_key)
    @source_object.copy_to(bucket: target_bucket.name, key: target_object_key)
    target_bucket.object(target_object_key)
  rescue Aws::Errors::ServiceError => e
```

```
    puts "Couldn't copy #{@source_object.key} to #{target_object_key}. Here's
 why: #{e.message}"
  end
end

# Example usage:
def run_demo
  source_bucket_name = "doc-example-bucket1"
  source_key = "my-source-file.txt"
  target_bucket_name = "doc-example-bucket2"
  target_key = "my-target-file.txt"

  source_bucket = Aws::S3::Bucket.new(source_bucket_name)
  wrapper = ObjectCopyWrapper.new(source_bucket.object(source_key))
  target_bucket = Aws::S3::Bucket.new(target_bucket_name)
  target_object = wrapper.copy_object(target_bucket, target_key)
  return unless target_object

  puts "Copied #{source_key} from #{source_bucket_name} to
 #{target_object.bucket_name}:#{target_object.key}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Salin objek dan tambahkan enkripsi di sisi server ke objek tujuan.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectCopyEncryptWrapper
  attr_reader :source_object

  # @param source_object [Aws::S3::Object] An existing Amazon S3 object. This is
 used as the source object for
  #                                        copy actions.
  def initialize(source_object)
    @source_object = source_object
  end

  # Copy the source object to the specified target bucket, rename it with the
 target key, and encrypt it.
  #
```

```
  # @param target_bucket [Aws::S3::Bucket] An existing Amazon S3 bucket where the
 object is copied.
  # @param target_object_key [String] The key to give the copy of the object.
  # @return [Aws::S3::Object, nil] The copied object when successful; otherwise,
 nil.
  def copy_object(target_bucket, target_object_key, encryption)
    @source_object.copy_to(bucket: target_bucket.name, key: target_object_key,
 server_side_encryption: encryption)
    target_bucket.object(target_object_key)
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't copy #{@source_object.key} to #{target_object_key}. Here's
 why: #{e.message}"
  end
end

# Example usage:
def run_demo
  source_bucket_name = "doc-example-bucket1"
  source_key = "my-source-file.txt"
  target_bucket_name = "doc-example-bucket2"
  target_key = "my-target-file.txt"
  target_encryption = "AES256"

  source_bucket = Aws::S3::Bucket.new(source_bucket_name)
  wrapper = ObjectCopyEncryptWrapper.new(source_bucket.object(source_key))
  target_bucket = Aws::S3::Bucket.new(target_bucket_name)
  target_object = wrapper.copy_object(target_bucket, target_key,
 target_encryption)
  return unless target_object

  puts "Copied #{source_key} from #{source_bucket_name} to
 #{target_object.bucket_name}:#{target_object.key} and "\
       "encrypted the target with #{target_object.server_side_encryption}
 encryption."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk detail API, lihat CopyObjectdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
pub async fn copy_object(
    client: &Client,
    bucket_name: &str,
    object_key: &str,
    target_key: &str,
) -> Result<CopyObjectOutput, SdkError<CopyObjectError>> {
    let mut source_bucket_and_object: String = "".to_owned();
    source_bucket_and_object.push_str(bucket_name);
    source_bucket_and_object.push('/');
    source_bucket_and_object.push_str(object_key);

    client
        .copy_object()
        .copy_source(source_bucket_and_object)
        .bucket(bucket_name)
        .key(target_key)
        .send()
        .await
}
```

- Untuk detail API, lihat CopyObjectreferensi AWS SDK for Rust API.

SAP ABAP

SDK untuk SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
TRY.
    lo_s3->copyobject(
      iv_bucket = iv_dest_bucket
      iv_key = iv_dest_object
      iv_copysource = |{ iv_src_bucket }/{ iv_src_object }|
    ).
    MESSAGE 'Object copied to another bucket.' TYPE 'I'.
  CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
  CATCH /aws1/cx_s3_nosuchkey.
    MESSAGE 'Object key does not exist.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat CopyObjectdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK untuk Swift

> **Note**
>
> Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    public func copyFile(from sourceBucket: String, name: String, to destBucket:
 String) async throws {
        let srcUrl = ("\(sourceBucket)/
\(name)").addingPercentEncoding(withAllowedCharacters: .urlPathAllowed)

        let input = CopyObjectInput(
            bucket: destBucket,
            copySource: srcUrl,
            key: name
        )
        _ = try await client.copyObject(input: input)
    }
```

- Untuk detail API, lihat referensi CopyObject AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateBucket`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateBucket`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai bucket dan objek
- Bekerja dengan objek berversi

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        /// <summary>
        /// Shows how to create a new Amazon S3 bucket.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client object.</param>
        /// <param name="bucketName">The name of the bucket to create.</param>
        /// <returns>A boolean value representing the success or failure of
        /// the bucket creation process.</returns>
        public static async Task<bool> CreateBucketAsync(IAmazonS3 client, string
bucketName)
        {
            try
            {
                var request = new PutBucketRequest
                {
                    BucketName = bucketName,
                    UseClientRegion = true,
                };

                var response = await client.PutBucketAsync(request);
                return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error creating bucket: '{ex.Message}'");
                return false;
            }
        }
```

Buat ember dengan kunci objek diaktifkan.

```
    /// <summary>
    /// Create a new Amazon S3 bucket with object lock actions.
    /// </summary>
    /// <param name="bucketName">The name of the bucket to create.</param>
    /// <param name="enableObjectLock">True to enable object lock on the
bucket.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> CreateBucketWithObjectLock(string bucketName, bool
enableObjectLock)
    {
        Console.WriteLine($"\tCreating bucket {bucketName} with object lock
{enableObjectLock}.");
        try
        {
            var request = new PutBucketRequest
            {
                BucketName = bucketName,
                UseClientRegion = true,
                ObjectLockEnabledForBucket = enableObjectLock,
            };

            var response = await _amazonS3.PutBucketAsync(request);

            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"Error creating bucket: '{ex.Message}'");
            return false;
        }
    }
```

- Untuk detail API, lihat CreateBucketdi Referensi AWS SDK for .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function create-bucket
#
# This function creates the specified bucket in the specified AWS Region, unless
# it already exists.
#
# Parameters:
#       -b bucket_name  -- The name of the bucket to create.
#       -r region_code  -- The code for an AWS Region in which to
#                          create the bucket.
#
```

```
# Returns:
#       The URL of the bucket that was created.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function create_bucket() {
  local bucket_name region_code response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function create_bucket"
    echo "Creates an Amazon S3 bucket. You must supply a bucket name:"
    echo "  -b bucket_name    The name of the bucket. It must be globally
 unique."
    echo "  [-r region_code]    The code for an AWS Region in which the bucket is
 created."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "b:r:h" option; do
    case "${option}" in
      b) bucket_name="${OPTARG}" ;;
      r) region_code="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  if [[ -z "$bucket_name" ]]; then
    errecho "ERROR: You must provide a bucket name with the -b parameter."
    usage
    return 1
  fi
```

```
  local bucket_config_arg
  # A location constraint for "us-east-1" returns an error.
  if [[ -n "$region_code" ]] && [[ "$region_code" != "us-east-1" ]]; then
    bucket_config_arg="--create-bucket-configuration LocationConstraint=
$region_code"
  fi

  iecho "Parameters:\n"
  iecho "    Bucket name:   $bucket_name"
  iecho "    Region code:   $region_code"
  iecho ""

  # If the bucket already exists, we don't want to try to create it.
  if (bucket_exists "$bucket_name"); then
    errecho "ERROR: A bucket with that name already exists. Try again."
    return 1
  fi

  # shellcheck disable=SC2086
  response=$(aws s3api create-bucket \
    --bucket "$bucket_name" \
    $bucket_config_arg)

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports create-bucket operation failed.\n$response"
    return 1
  fi
}
```

- Untuk detail API, lihat CreateBucketdi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::createBucket(const Aws::String &bucketName,
                              const Aws::S3::S3ClientConfiguration &clientConfig)
 {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::CreateBucketRequest request;
    request.SetBucket(bucketName);

    if (clientConfig.region != "us-east-1") {
        Aws::S3::Model::CreateBucketConfiguration createBucketConfig;
        createBucketConfig.SetLocationConstraint(

 Aws::S3::Model::BucketLocationConstraintMapper::GetBucketLocationConstraintForName(
                        clientConfig.region));
        request.SetCreateBucketConfiguration(createBucketConfig);
    }

    Aws::S3::Model::CreateBucketOutcome outcome = client.CreateBucket(request);
    if (!outcome.IsSuccess()) {
        auto err = outcome.GetError();
        std::cerr << "Error: createBucket: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    } else {
        std::cout << "Created bucket " << bucketName <<
                  " in the specified AWS Region." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat CreateBucketdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Contoh 1: Untuk membuat ember

`create-bucket`Contoh berikut membuat bucket bernama`my-bucket`:

```
aws s3api create-bucket \
```

```
    --bucket my-bucket \
    --region us-east-1
```

Output:

```
{
    "Location": "/my-bucket"
}
```

Lihat informasi yang lebih lengkap di Membuat bucket dalam Panduan Pengguna Amazon S3.

Contoh 2: Untuk membuat ember dengan pemilik diberlakukan

`create-bucket`Contoh berikut membuat bucket bernama yang menggunakan setelan `my-bucket` yang diterapkan pemilik bucket untuk Kepemilikan Objek S3.

```
aws s3api create-bucket \
    --bucket my-bucket \
    --region us-east-1 \
    --object-ownership BucketOwnerEnforced
```

Output:

```
{
    "Location": "/my-bucket"
}
```

Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL di Panduan Pengguna Amazon S3.

Contoh 3: Untuk membuat bucket di luar wilayah ``us-east-1``

`create-bucket`Contoh berikut membuat bucket bernama `my-bucket` di `eu-west-1` wilayah tersebut. Daerah di luar `us-east-1` memerlukan yang sesuai `LocationConstraint` untuk ditentukan untuk membuat ember di wilayah yang diinginkan.

```
aws s3api create-bucket \
    --bucket my-bucket \
    --region eu-west-1 \
    --create-bucket-configuration LocationConstraint=eu-west-1
```

Output:

```
{
    "Location": "http://my-bucket.s3.amazonaws.com/"
}
```

Lihat informasi yang lebih lengkap di Membuat bucket dalam Panduan Pengguna Amazon S3.

- Untuk detail API, lihat CreateBucketdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat bucket dengan konfigurasi default.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// CreateBucket creates a bucket with the specified name in the specified Region.
func (basics BucketBasics) CreateBucket(name string, region string) error {
 _, err := basics.S3Client.CreateBucket(context.TODO(), &s3.CreateBucketInput{
  Bucket: aws.String(name),
  CreateBucketConfiguration: &types.CreateBucketConfiguration{
   LocationConstraint: types.BucketLocationConstraint(region),
  },
```

```
 })
 if err != nil {
  log.Printf("Couldn't create bucket %v in Region %v. Here's why: %v\n",
   name, region, err)
 }
 return err
}
```

Buat ember dengan penguncian objek dan tunggu sampai ada.

```
// S3Actions wraps S3 service actions.
type S3Actions struct {
 S3Client  *s3.Client
 S3Manager *manager.Uploader
}



// CreateBucketWithLock creates a new S3 bucket with optional object locking
 enabled
// and waits for the bucket to exist before returning.
func (actor S3Actions) CreateBucketWithLock(ctx context.Context, bucket string,
 region string, enableObjectLock bool) (string, error) {
 input := &s3.CreateBucketInput{
  Bucket: aws.String(bucket),
  CreateBucketConfiguration: &types.CreateBucketConfiguration{
   LocationConstraint: types.BucketLocationConstraint(region),
  },
 }

 if enableObjectLock {
  input.ObjectLockEnabledForBucket = aws.Bool(true)
 }

 _, err := actor.S3Client.CreateBucket(ctx, input)
 if err != nil {
  var owned *types.BucketAlreadyOwnedByYou
  var exists *types.BucketAlreadyExists
  if errors.As(err, &owned) {
   log.Printf("You already own bucket %s.\n", bucket)
```

```
    err = owned
   } else if errors.As(err, &exists) {
    log.Printf("Bucket %s already exists.\n", bucket)
    err = exists
   }
  } else {
   err = s3.NewBucketExistsWaiter(actor.S3Client).Wait(
    ctx, &s3.HeadBucketInput{Bucket: aws.String(bucket)}, time.Minute)
   if err != nil {
    log.Printf("Failed attempt to wait for bucket %s to exist.\n", bucket)
   }
  }

  return bucket, err
}
```

- Untuk detail API, lihat CreateBucketdi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat bucket.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.CreateBucketRequest;
import software.amazon.awssdk.services.s3.model.HeadBucketRequest;
import software.amazon.awssdk.services.s3.model.HeadBucketResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.waiters.S3Waiter;
import java.net.URISyntaxException;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class CreateBucket {
    public static void main(String[] args) throws URISyntaxException {
        final String usage = """

                Usage:
                    <bucketName>\s

                Where:
                    bucketName - The name of the bucket to create. The bucket
 name must be unique, or an error occurs.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        System.out.format("Creating a bucket named %s\n", bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        createBucket(s3, bucketName);
        s3.close();
    }

    public static void createBucket(S3Client s3Client, String bucketName) {
        try {
            S3Waiter s3Waiter = s3Client.waiter();
            CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
                    .bucket(bucketName)
                    .build();
```

```
            s3Client.createBucket(bucketRequest);
            HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
                    .bucket(bucketName)
                    .build();

            // Wait until the bucket is created and print out the response.
            WaiterResponse<HeadBucketResponse> waiterResponse =
 s3Waiter.waitUntilBucketExists(bucketRequestWait);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println(bucketName + " is ready");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

Buat ember dengan kunci objek diaktifkan.

```
    // Create a new Amazon S3 bucket with object lock options.
    public void createBucketWithLockOptions(boolean enableObjectLock, String
 bucketName) {
        S3Waiter s3Waiter = getClient().waiter();
        CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
            .bucket(bucketName)
            .objectLockEnabledForBucket(enableObjectLock)
            .build();

        getClient().createBucket(bucketRequest);
        HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
            .bucket(bucketName)
            .build();

        // Wait until the bucket is created and print out the response.
        s3Waiter.waitUntilBucketExists(bucketRequestWait);
        System.out.println(bucketName + " is ready");
    }
```

- Untuk detail API, lihat CreateBucketdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat bucket.

```
import { CreateBucketCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new CreateBucketCommand({
    // The name of the bucket. Bucket names are unique and have several other
 constraints.
    // See https://docs.aws.amazon.com/AmazonS3/latest/userguide/
bucketnamingrules.html
    Bucket: "bucket-name",
  });

  try {
    const { Location } = await client.send(command);
    console.log(`Bucket created with location ${Location}`);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat CreateBucketdi Referensi AWS SDK for JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun createNewBucket(bucketName: String) {
    val request =
        CreateBucketRequest {
            bucket = bucketName
        }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.createBucket(request)
        println("$bucketName is ready")
    }
}
```

- Untuk detail API, lihat CreateBucketdi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat bucket.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        try {
```

```
            $this->s3client->createBucket([
                'Bucket' => $this->bucketName,
                'CreateBucketConfiguration' => ['LocationConstraint' => $region],
            ]);
            echo "Created bucket named: $this->bucketName \n";
        } catch (Exception $exception) {
            echo "Failed to create bucket $this->bucketName with error: " .
 $exception->getMessage();
            exit("Please fix error with bucket creation before continuing.");
        }
```

- Untuk detail API, lihat CreateBucketdi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat bucket dengan pengaturan default.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def create(self, region_override=None):
        """
```

```
        Create an Amazon S3 bucket in the default Region for the account or in
 the
        specified Region.

        :param region_override: The Region in which to create the bucket. If this
 is
                                not specified, the Region configured in your
 shared
                                credentials is used.
        """
        if region_override is not None:
            region = region_override
        else:
            region = self.bucket.meta.client.meta.region_name
        try:
            self.bucket.create(CreateBucketConfiguration={"LocationConstraint":
 region})

            self.bucket.wait_until_exists()
            logger.info("Created bucket '%s' in region=%s", self.bucket.name,
 region)
        except ClientError as error:
            logger.exception(
                "Couldn't create bucket named '%s' in region=%s.",
                self.bucket.name,
                region,
            )
            raise error
```

Buat bucket berversi dengan konfigurasi siklus hidup.

```
def create_versioned_bucket(bucket_name, prefix):
    """
    Creates an Amazon S3 bucket, enables it for versioning, and configures a
 lifecycle
    that expires noncurrent object versions after 7 days.

    Adding a lifecycle configuration to a versioned bucket is a best practice.
    It helps prevent objects in the bucket from accumulating a large number of
    noncurrent versions, which can slow down request performance.
```

```
    Usage is shown in the usage_demo_single_object function at the end of this
module.

    :param bucket_name: The name of the bucket to create.
    :param prefix: Identifies which objects are automatically expired under the
                   configured lifecycle rules.
    :return: The newly created bucket.
    """
    try:
        bucket = s3.create_bucket(
            Bucket=bucket_name,
            CreateBucketConfiguration={
                "LocationConstraint": s3.meta.client.meta.region_name
            },
        )
        logger.info("Created bucket %s.", bucket.name)
    except ClientError as error:
        if error.response["Error"]["Code"] == "BucketAlreadyOwnedByYou":
            logger.warning("Bucket %s already exists! Using it.", bucket_name)
            bucket = s3.Bucket(bucket_name)
        else:
            logger.exception("Couldn't create bucket %s.", bucket_name)
            raise

    try:
        bucket.Versioning().enable()
        logger.info("Enabled versioning on bucket %s.", bucket.name)
    except ClientError:
        logger.exception("Couldn't enable versioning on bucket %s.", bucket.name)
        raise

    try:
        expiration = 7
        bucket.LifecycleConfiguration().put(
            LifecycleConfiguration={
                "Rules": [
                    {
                        "Status": "Enabled",
                        "Prefix": prefix,
                        "NoncurrentVersionExpiration": {"NoncurrentDays":
expiration},
                    }
                ]
            }
```

```
        )
        logger.info(
            "Configured lifecycle to expire noncurrent versions after %s days "
            "on bucket %s.",
            expiration,
            bucket.name,
        )
    except ClientError as error:
        logger.warning(
            "Couldn't configure lifecycle on bucket %s because %s. "
            "Continuing anyway.",
            bucket.name,
            error,
        )

    return bucket
```

- Untuk detail API, lihat CreateBucketdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket actions.
class BucketCreateWrapper
  attr_reader :bucket

  # @param bucket [Aws::S3::Bucket] An Amazon S3 bucket initialized with a name.
 This is a client-side object until
  #                                 create is called.
  def initialize(bucket)
```

```
    @bucket = bucket
  end

  # Creates an Amazon S3 bucket in the specified AWS Region.
  #
  # @param region [String] The Region where the bucket is created.
  # @return [Boolean] True when the bucket is created; otherwise, false.
  def create?(region)
    @bucket.create(create_bucket_configuration: { location_constraint: region })
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't create bucket. Here's why: #{e.message}"
    false
  end

  # Gets the Region where the bucket is located.
  #
  # @return [String] The location of the bucket.
  def location
    if @bucket.nil?
      "None. You must create a bucket before you can get its location!"
    else
      @bucket.client.get_bucket_location(bucket:
 @bucket.name).location_constraint
    end
  rescue Aws::Errors::ServiceError => e
    "Couldn't get the location of #{@bucket.name}. Here's why: #{e.message}"
  end
end

# Example usage:
def run_demo
  region = "us-west-2"
  wrapper = BucketCreateWrapper.new(Aws::S3::Bucket.new("doc-example-bucket-
#{Random.uuid}"))
  return unless wrapper.create?(region)

  puts "Created bucket #{wrapper.bucket.name}."
  puts "Your bucket's region is: #{wrapper.location}"
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk detail API, lihat CreateBucketdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
pub async fn create_bucket(
    client: &Client,
    bucket_name: &str,
    region: &str,
) -> Result<CreateBucketOutput, SdkError<CreateBucketError>> {
    let constraint = BucketLocationConstraint::from(region);
    let cfg = CreateBucketConfiguration::builder()
        .location_constraint(constraint)
        .build();
    client
        .create_bucket()
        .create_bucket_configuration(cfg)
        .bucket(bucket_name)
        .send()
        .await
}
```

- Untuk detail API, lihat CreateBucketreferensi AWS SDK for Rust API.

SAP ABAP

SDK untuk SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
TRY.
    lo_s3->createbucket(
        iv_bucket = iv_bucket_name
    ).
    MESSAGE 'S3 bucket created.' TYPE 'I'.
  CATCH /aws1/cx_s3_bucketalrdyexists.
    MESSAGE 'Bucket name already exists.' TYPE 'E'.
  CATCH /aws1/cx_s3_bktalrdyownedbyyou.
    MESSAGE 'Bucket already exists and is owned by you.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat CreateBucketdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK untuk Swift

> **Note**
>
> Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    public func createBucket(name: String) async throws {
        let config = S3ClientTypes.CreateBucketConfiguration(
            locationConstraint: .usEast2
        )
        let input = CreateBucketInput(
            bucket: name,
            createBucketConfiguration: config
        )
        _ = try await client.createBucket(input: input)
    }
```

- Untuk detail API, lihat referensi CreateBucket AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateMultiRegionAccessPoint`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateMultiRegionAccessPoint`.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Konfigurasikan klien kontrol S3 untuk mengirim permintaan ke Wilayah us-west-2.

```
        suspend fun createS3ControlClient(): S3ControlClient {
            // Configure your S3ControlClient to send requests to US West
(Oregon).
            val s3Control = S3ControlClient.fromEnvironment {
                region = "us-west-2"
            }
```

```
            return s3Control
        }
```

Buat Titik Akses Multi-Wilayah.

```
    suspend fun createMrap(
        s3Control: S3ControlClient,
        accountIdParam: String,
        bucketName1: String,
        bucketName2: String,
        mrapName: String,
    ): String {
        println("Creating MRAP ...")
        val createMrapResponse: CreateMultiRegionAccessPointResponse =
            s3Control.createMultiRegionAccessPoint {
                accountId = accountIdParam
                clientToken = UUID.randomUUID().toString()
                details {
                    name = mrapName
                    regions = listOf(
                        Region {
                            bucket = bucketName1
                        },
                        Region {
                            bucket = bucketName2
                        },
                    )
                }
            }
        val requestToken: String? = createMrapResponse.requestTokenArn

        // Use the request token to check for the status of the
 CreateMultiRegionAccessPoint operation.
        if (requestToken != null) {
            waitForSucceededStatus(s3Control, requestToken, accountIdParam)
            println("MRAP created")
        }

        val getMrapResponse =
            s3Control.getMultiRegionAccessPoint(
                input = GetMultiRegionAccessPointRequest {
                    accountId = accountIdParam
```

```
                    name = mrapName
                },
            )
        val mrapAlias = getMrapResponse.accessPoint?.alias
        return "arn:aws:s3::$accountIdParam:accesspoint/$mrapAlias"
    }
```

Tunggu hingga Titik Akses Multi-Wilayah tersedia.

```
        suspend fun waitForSucceededStatus(
            s3Control: S3ControlClient,
            requestToken: String,
            accountIdParam: String,
            timeBetweenChecks: Duration = 1.minutes,
        ) {
            var describeResponse: DescribeMultiRegionAccessPointOperationResponse
            describeResponse = s3Control.describeMultiRegionAccessPointOperation(
                input = DescribeMultiRegionAccessPointOperationRequest {
                    accountId = accountIdParam
                    requestTokenArn = requestToken
                },
            )

            var status: String? = describeResponse.asyncOperation?.requestStatus
            while (status != "SUCCEEDED") {
                delay(timeBetweenChecks)
                describeResponse =
 s3Control.describeMultiRegionAccessPointOperation(
                    input = DescribeMultiRegionAccessPointOperationRequest {
                        accountId = accountIdParam
                        requestTokenArn = requestToken
                    },
                )
                status = describeResponse.asyncOperation?.requestStatus
                println(status)
            }
        }
```

- Untuk informasi selengkapnya, lihat AWS panduan pengembang SDK untuk Kotlin.

- Untuk detail API, lihat CreateMultiRegionAccessPointdi AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateMultipartUpload`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateMultipartUpload`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Melakukan penyalinan multibagian
- Melakukan pengunggahan multibagian
- Gunakan checksum

CLI

AWS CLI

Perintah berikut membuat unggahan multipart di bucket `my-bucket` dengan kunci`multipart/01`:

```
aws s3api create-multipart-upload --bucket my-bucket --key 'multipart/01'
```

Output:

```
{
    "Bucket": "my-bucket",
    "UploadId":
 "dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3
    "Key": "multipart/01"
}
```

File yang sudah selesai akan diberi nama `01` dalam folder yang disebut `multipart` dalam ember`my-bucket`. Simpan ID unggahan, kunci, dan nama bucket untuk digunakan dengan `upload-part` perintah.

- Untuk detail API, lihat CreateMultipartUploaddi Referensi AWS CLI Perintah.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
let multipart_upload_res: CreateMultipartUploadOutput = client
    .create_multipart_upload()
    .bucket(&bucket_name)
    .key(&key)
    .send()
    .await
    .unwrap();
```

- Untuk detail API, lihat CreateMultipartUploadreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan `DeleteBucket` dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteBucket`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai bucket dan objek

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        /// <summary>
        /// Shows how to delete an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client object.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket to
delete.</param>
        /// <returns>A boolean value that represents the success or failure of
        /// the delete operation.</returns>
        public static async Task<bool> DeleteBucketAsync(IAmazonS3 client, string
bucketName)
        {
            var request = new DeleteBucketRequest
            {
                BucketName = bucketName,
            };

            var response = await client.DeleteBucketAsync(request);
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
```

- Untuk detail API, lihat DeleteBucketdi Referensi AWS SDK for .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function delete_bucket
#
# This function deletes the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.

# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function delete_bucket() {
  local bucket_name=$1
  local response

  response=$(aws s3api delete-bucket \
    --bucket "$bucket_name")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR: AWS reports s3api delete-bucket failed.\n$response"
    return 1
```

```
  fi
}
```

- Untuk detail API, lihat DeleteBucketdi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::deleteBucket(const Aws::String &bucketName,
                              const Aws::S3::S3ClientConfiguration &clientConfig)
 {

    Aws::S3::S3Client client(clientConfig);

    Aws::S3::Model::DeleteBucketRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::DeleteBucketOutcome outcome =
            client.DeleteBucket(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: deleteBucket: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    } else {
        std::cout << "The bucket was deleted" << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat DeleteBucketdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Perintah berikut menghapus bucket bernamamy-bucket:

```
aws s3api delete-bucket --bucket my-bucket --region us-east-1
```

- Untuk detail API, lihat DeleteBucketdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// DeleteBucket deletes a bucket. The bucket must be empty or an error is
 returned.
func (basics BucketBasics) DeleteBucket(bucketName string) error {
 _, err := basics.S3Client.DeleteBucket(context.TODO(), &s3.DeleteBucketInput{
  Bucket: aws.String(bucketName)})
 if err != nil {
```

```
  log.Printf("Couldn't delete bucket %v. Here's why: %v\n", bucketName, err)
 }
 return err
}
```

- Untuk detail API, lihat DeleteBucketdi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        DeleteBucketRequest deleteBucketRequest = DeleteBucketRequest.builder()
                .bucket(bucket)
                .build();

        s3.deleteBucket(deleteBucketRequest);
        s3.close();
```

- Untuk detail API, lihat DeleteBucketdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Hapus bucket.

```
import { DeleteBucketCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

// Delete a bucket.
export const main = async () => {
  const command = new DeleteBucketCommand({
    Bucket: "test-bucket",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat DeleteBucketdi Referensi AWS SDK for JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Hapus bucket kosong.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        try {
            $this->s3client->deleteBucket([
                'Bucket' => $this->bucketName,
            ]);
            echo "Deleted bucket $this->bucketName.\n";
```

```
        } catch (Exception $exception) {
            echo "Failed to delete $this->bucketName with error: " . $exception-
>getMessage();
            exit("Please fix error with bucket deletion before continuing.");
        }
```

- Untuk detail API, lihat DeleteBucketdi Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menghapus semua objek dan versi objek dari bucket 'test-files' dan kemudian menghapus bucket. Perintah akan meminta konfirmasi sebelum melanjutkan. Tambahkan sakelar -Force untuk menekan konfirmasi. Perhatikan bahwa ember yang tidak kosong tidak dapat dihapus.

```
Remove-S3Bucket -BucketName test-files -DeleteBucketContent
```

- Untuk detail API, lihat DeleteBucketdi Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
```

```
        """
        self.bucket = bucket
        self.name = bucket.name


    def delete(self):
        """
        Delete the bucket. The bucket must be empty or an error is raised.
        """
        try:
            self.bucket.delete()
            self.bucket.wait_until_not_exists()
            logger.info("Bucket %s successfully deleted.", self.bucket.name)
        except ClientError:
            logger.exception("Couldn't delete bucket %s.", self.bucket.name)
            raise
```

- Untuk detail API, lihat DeleteBucketdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
 # Deletes the objects in an Amazon S3 bucket and deletes the bucket.
 #
 # @param bucket [Aws::S3::Bucket] The bucket to empty and delete.
 def delete_bucket(bucket)
   puts("\nDo you want to delete all of the objects as well as the bucket (y/n)?
")
   answer = gets.chomp.downcase
   if answer == "y"
     bucket.objects.batch_delete!
     bucket.delete
     puts("Emptied and deleted bucket #{bucket.name}.\n")
```

```
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't empty and delete bucket #{bucket.name}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  end
```

- Untuk detail API, lihat DeleteBucketdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
pub async fn delete_bucket(client: &Client, bucket_name: &str) -> Result<(),
 Error> {
    client.delete_bucket().bucket(bucket_name).send().await?;
    println!("Bucket deleted");
    Ok(())
}
```

- Untuk detail API, lihat DeleteBucketreferensi AWS SDK for Rust API.

SAP ABAP

SDK untuk SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    TRY.

        lo_s3->deletebucket(
            iv_bucket = iv_bucket_name
        ).
        MESSAGE 'Deleted S3 bucket.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat DeleteBucketdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK untuk Swift

> **Note**
>
> Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    public func deleteBucket(name: String) async throws {
        let input = DeleteBucketInput(
            bucket: name
        )
        _ = try await client.deleteBucket(input: input)
    }
```

- Untuk detail API, lihat referensi DeleteBucket AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteBucketAnalyticsConfiguration`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteBucketAnalyticsConfiguration`.

CLI

AWS CLI

Untuk menghapus konfigurasi analitik untuk bucket

`delete-bucket-analytics-configuration`Contoh berikut menghapus konfigurasi analitik untuk bucket dan ID yang ditentukan.

```
aws s3api delete-bucket-analytics-configuration \
    --bucket my-bucket \
    --id 1
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat DeleteBucketAnalyticsConfigurationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah menghapus filter analitik dengan nama 'testfilter' di bucket S3 yang diberikan.

```
Remove-S3BucketAnalyticsConfiguration -BucketName 's3testbucket' -AnalyticsId
 'testfilter'
```

- Untuk detail API, lihat DeleteBucketAnalyticsConfigurationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteBucketCors`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteBucketCors`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        /// <summary>
        /// Deletes a CORS configuration from an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used
        /// to delete the CORS configuration from the bucket.</param>
        private static async Task DeleteCORSConfigurationAsync(AmazonS3Client
client)
        {
            DeleteCORSConfigurationRequest request = new
DeleteCORSConfigurationRequest()
            {
                BucketName = BucketName,
            };
            await client.DeleteCORSConfigurationAsync(request);
        }
```

- Untuk detail API, lihat DeleteBucketCorsdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Perintah berikut menghapus konfigurasi Cross-Origin Resource Sharing dari bucket bernama: `my-bucket`

```
aws s3api delete-bucket-cors --bucket my-bucket
```

- Untuk detail API, lihat DeleteBucketCorsdi Referensi AWS CLI Perintah.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def delete_cors(self):
        """
        Delete the CORS rules from the bucket.

        :param bucket_name: The name of the bucket to update.
        """
        try:
```

```
            self.bucket.Cors().delete()
            logger.info("Deleted CORS from bucket '%s'.", self.bucket.name)
        except ClientError:
            logger.exception("Couldn't delete CORS from bucket '%s'.",
 self.bucket.name)
            raise
```

- Untuk detail API, lihat DeleteBucketCorsdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket CORS configuration.
class BucketCorsWrapper
  attr_reader :bucket_cors

  # @param bucket_cors [Aws::S3::BucketCors] A bucket CORS object configured with
 an existing bucket.
  def initialize(bucket_cors)
    @bucket_cors = bucket_cors
  end

  # Deletes the CORS configuration of a bucket.
  #
  # @return [Boolean] True if the CORS rules were deleted; otherwise, false.
  def delete_cors
    @bucket_cors.delete
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't delete CORS rules for #{@bucket_cors.bucket.name}. Here's why:
 #{e.message}"
```

```
    false
  end

end
```

- Untuk detail API, lihat DeleteBucketCorsdi Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteBucketEncryption`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteBucketEncryption`.

CLI

AWS CLI

Untuk menghapus konfigurasi enkripsi sisi server dari bucket

`delete-bucket-encryption`Contoh berikut menghapus konfigurasi enkripsi sisi server dari bucket yang ditentukan.

```
aws s3api delete-bucket-encryption \
    --bucket my-bucket
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat DeleteBucketEncryptiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Ini menonaktifkan enkripsi yang diaktifkan untuk bucket S3 yang disediakan.

```
Remove-S3BucketEncryption -BucketName 's3casetestbucket'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketEncryption (DeleteBucketEncryption)" on
 target "s3casetestbucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- Untuk detail API, lihat DeleteBucketEncryptiondi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteBucketInventoryConfiguration`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteBucketInventoryConfiguration`.

CLI

AWS CLI

Untuk menghapus konfigurasi inventaris bucket

`delete-bucket-inventory-configuration`Contoh berikut menghapus konfigurasi inventaris dengan ID 1 untuk bucket yang ditentukan.

```
aws s3api delete-bucket-inventory-configuration \
    --bucket my-bucket \
    --id 1
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat DeleteBucketInventoryConfigurationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menghapus invventori bernama 'testInventoryName' yang sesuai dengan bucket S3 yang diberikan.

```
Remove-S3BucketInventoryConfiguration -BucketName 's3testbucket' -InventoryId
 'testInventoryName'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketInventoryConfiguration
 (DeleteBucketInventoryConfiguration)" on target "s3testbucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- Untuk detail API, lihat DeleteBucketInventoryConfigurationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteBucketLifecycle`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteBucketLifecycle`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        /// <summary>
        /// This method removes the Lifecycle configuration from the named
        /// S3 bucket.
        /// </summary>
        /// <param name="client">The S3 client object used to call
        /// the RemoveLifecycleConfigAsync method.</param>
        /// <param name="bucketName">A string representing the name of the
        /// S3 bucket from which the configuration will be removed.</param>
        public static async Task RemoveLifecycleConfigAsync(IAmazonS3 client,
 string bucketName)
        {
            var request = new DeleteLifecycleConfigurationRequest()
            {
                BucketName = bucketName,
            };
            await client.DeleteLifecycleConfigurationAsync(request);
        }
```

- Untuk detail API, lihat DeleteBucketLifecycledi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Perintah berikut menghapus konfigurasi siklus hidup dari bucket bernama: `my-bucket`

```
aws s3api delete-bucket-lifecycle --bucket my-bucket
```

- Untuk detail API, lihat DeleteBucketLifecycledi Referensi AWS CLI Perintah.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def delete_lifecycle_configuration(self):
        """
        Remove the lifecycle configuration from the specified bucket.
        """
        try:
            self.bucket.LifecycleConfiguration().delete()
            logger.info(
                "Deleted lifecycle configuration for bucket '%s'.",
 self.bucket.name
            )
        except ClientError:
            logger.exception(
                "Couldn't delete lifecycle configuration for bucket '%s'.",
                self.bucket.name,
            )
            raise
```

- Untuk detail API, lihat DeleteBucketLifecycledi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteBucketMetricsConfiguration`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteBucketMetricsConfiguration`.

CLI

AWS CLI

Untuk menghapus konfigurasi metrik untuk bucket

`delete-bucket-metrics-configuration`Contoh berikut menghapus konfigurasi metrik untuk bucket dan ID yang ditentukan.

```
aws s3api delete-bucket-metrics-configuration \
    --bucket my-bucket \
    --id 123
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat DeleteBucketMetricsConfigurationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah menghapus filter metrik dengan nama 'testmetrics' di bucket S3 yang diberikan.

```
Remove-S3BucketMetricsConfiguration -BucketName 's3testbucket' -MetricsId
 'testmetrics'
```

- Untuk detail API, lihat DeleteBucketMetricsConfigurationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteBucketPolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteBucketPolicy`.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::deleteBucketPolicy(const Aws::String &bucketName,
                                    const Aws::S3::S3ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client client(clientConfig);

    Aws::S3::Model::DeleteBucketPolicyRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::DeleteBucketPolicyOutcome outcome =
 client.DeleteBucketPolicy(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: deleteBucketPolicy: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    } else {
        std::cout << "Policy was deleted from the bucket." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat DeleteBucketPolicydi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Perintah berikut menghapus kebijakan bucket dari bucket bernamamy-bucket:

```
aws s3api delete-bucket-policy --bucket my-bucket
```

- Untuk detail API, lihat DeleteBucketPolicydi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.DeleteBucketPolicyRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class DeleteBucketPolicy {
    public static void main(String[] args) {

        final String usage = """
```

```
                Usage:
                    <bucketName>

                Where:
                    bucketName - The Amazon S3 bucket to delete the policy from
 (for example, bucket1).""";

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        System.out.format("Deleting policy from bucket: \"%s\"\n\n", bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        deleteS3BucketPolicy(s3, bucketName);
        s3.close();
    }

    // Delete the bucket policy.
    public static void deleteS3BucketPolicy(S3Client s3, String bucketName) {
        DeleteBucketPolicyRequest delReq = DeleteBucketPolicyRequest.builder()
                .bucket(bucketName)
                .build();

        try {
            s3.deleteBucketPolicy(delReq);
            System.out.println("Done!");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat DeleteBucketPolicydi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Hapus kebijakan bucket.

```
import { DeleteBucketPolicyCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

// This will remove the policy from the bucket.
export const main = async () => {
  const command = new DeleteBucketPolicyCommand({
    Bucket: "test-bucket",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat DeleteBucketPolicydi Referensi AWS SDK for JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun deleteS3BucketPolicy(bucketName: String?) {
    val request =
        DeleteBucketPolicyRequest {
            bucket = bucketName
        }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.deleteBucketPolicy(request)
        println("Done!")
    }
}
```

- Untuk detail API, lihat DeleteBucketPolicydi AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah menghapus kebijakan bucket yang terkait dengan bucket S3 yang diberikan.

```
Remove-S3BucketPolicy -BucketName 's3testbucket'
```

- Untuk detail API, lihat DeleteBucketPolicydi Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def delete_policy(self):
        """
        Delete the security policy from the bucket.
        """
        try:
            self.bucket.Policy().delete()
            logger.info("Deleted policy for bucket '%s'.", self.bucket.name)
        except ClientError:
            logger.exception(
                "Couldn't delete policy for bucket '%s'.", self.bucket.name
            )
            raise
```

- Untuk detail API, lihat DeleteBucketPolicydi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# Wraps an Amazon S3 bucket policy.
class BucketPolicyWrapper
  attr_reader :bucket_policy

  # @param bucket_policy [Aws::S3::BucketPolicy] A bucket policy object
 configured with an existing bucket.
  def initialize(bucket_policy)
    @bucket_policy = bucket_policy
  end

  def delete_policy
    @bucket_policy.delete
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't delete the policy from #{@bucket_policy.bucket.name}. Here's
 why: #{e.message}"
    false
  end

end
```

- Untuk detail API, lihat DeleteBucketPolicydi Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteBucketReplication`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteBucketReplication`.

CLI

AWS CLI

Perintah berikut menghapus konfigurasi replikasi dari bucket bernama: `my-bucket`

```
aws s3api delete-bucket-replication --bucket my-bucket
```

- Untuk detail API, lihat DeleteBucketReplicationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Menghapus konfigurasi replikasi yang terkait dengan bucket bernama 'mybucket'. Perhatikan bahwa operasi ini memerlukan izin untuk DeleteReplicationConfiguration tindakan s3:. Anda akan diminta konfirmasi sebelum operasi berlangsung - untuk menekan konfirmasi, gunakan sakelar -Force.

```
Remove-S3BucketReplication -BucketName mybucket
```

- Untuk detail API, lihat DeleteBucketReplicationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteBucketTagging`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteBucketTagging`.

CLI

AWS CLI

Perintah berikut menghapus konfigurasi penandaan dari bucket bernama: `my-bucket`

```
aws s3api delete-bucket-tagging --bucket my-bucket
```

- Untuk detail API, lihat DeleteBucketTaggingdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menghapus semua tag yang terkait dengan bucket S3 yang diberikan.

```
Remove-S3BucketTagging -BucketName 's3testbucket'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketTagging (DeleteBucketTagging)" on target
 "s3testbucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- Untuk detail API, lihat DeleteBucketTaggingdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan `DeleteBucketWebsite` dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteBucketWebsite`.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::deleteBucketWebsite(const Aws::String &bucketName,
```

```
                                        const Aws::S3::S3ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::DeleteBucketWebsiteRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::DeleteBucketWebsiteOutcome outcome =
            client.DeleteBucketWebsite(request);

    if (!outcome.IsSuccess()) {
        auto err = outcome.GetError();
        std::cerr << "Error: deleteBucketWebsite: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    } else {
        std::cout << "Website configuration was removed." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat DeleteBucketWebsitedi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Perintah berikut menghapus konfigurasi situs web dari bucket bernamamy-bucket:

```
aws s3api delete-bucket-website --bucket my-bucket
```

- Untuk detail API, lihat DeleteBucketWebsitedi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.DeleteBucketWebsiteRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class DeleteWebsiteConfiguration {
    public static void main(String[] args) {
        final String usage = """

                Usage:     <bucketName>

                Where:
                    bucketName - The Amazon S3 bucket to delete the website
 configuration from.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
```

```
        System.out.format("Deleting website configuration for Amazon S3 bucket:
 %s\n", bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        deleteBucketWebsiteConfig(s3, bucketName);
        System.out.println("Done!");
        s3.close();
    }

    public static void deleteBucketWebsiteConfig(S3Client s3, String bucketName)
 {
        DeleteBucketWebsiteRequest delReq = DeleteBucketWebsiteRequest.builder()
                .bucket(bucketName)
                .build();

        try {
            s3.deleteBucketWebsite(delReq);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.out.println("Failed to delete website configuration!");
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat DeleteBucketWebsitedi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Hapus konfigurasi situs web dari bucket.

```
import { DeleteBucketWebsiteCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

// Disable static website hosting on the bucket.
export const main = async () => {
  const command = new DeleteBucketWebsiteCommand({
    Bucket: "test-bucket",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat DeleteBucketWebsitedi Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menonaktifkan properti hosting situs web statis dari bucket S3 yang diberikan.

```
Remove-S3BucketWebsite -BucketName 's3testbucket'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3BucketWebsite (DeleteBucketWebsite)" on target
 "s3testbucket".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- Untuk detail API, lihat DeleteBucketWebsitedi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan `DeleteObject` dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteObject`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Bekerja dengan objek berversi

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Hapus objek dalam bucket S3 yang tidak berversi.

```
using System;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to delete an object from a non-versioned Amazon
/// Simple Storage Service (Amazon S3) bucket.
/// </summary>
public class DeleteObject
{
    /// <summary>
    /// The Main method initializes the necessary variables and then calls
    /// the DeleteObjectNonVersionedBucketAsync method to delete the object
```

```
        /// named by the keyName parameter.
        /// </summary>
        public static async Task Main()
        {
            const string bucketName = "doc-example-bucket";
            const string keyName = "testfile.txt";

            // If the Amazon S3 bucket is located in an AWS Region other than the
            // Region of the default account, define the AWS Region for the
            // Amazon S3 bucket in your call to the AmazonS3Client constructor.
            // For example RegionEndpoint.USWest2.
            IAmazonS3 client = new AmazonS3Client();
            await DeleteObjectNonVersionedBucketAsync(client, bucketName,
keyName);
        }

        /// <summary>
        /// The DeleteObjectNonVersionedBucketAsync takes care of deleting the
        /// desired object from the named bucket.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client used to delete
        /// an object from an Amazon S3 bucket.</param>
        /// <param name="bucketName">The name of the bucket from which the
        /// object will be deleted.</param>
        /// <param name="keyName">The name of the object to delete.</param>
        public static async Task DeleteObjectNonVersionedBucketAsync(IAmazonS3
client, string bucketName, string keyName)
        {
            try
            {
                var deleteObjectRequest = new DeleteObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                };

                Console.WriteLine($"Deleting object: {keyName}");
                await client.DeleteObjectAsync(deleteObjectRequest);
                Console.WriteLine($"Object: {keyName} deleted from
{bucketName}.");
            }
            catch (AmazonS3Exception ex)
            {
```

```
                Console.WriteLine($"Error encountered on server.
Message:'{ex.Message}' when deleting an object.");
            }
        }
    }
```

Hapus objek dalam bucket S3 berversi.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example creates an object in an Amazon Simple Storage Service
    /// (Amazon S3) bucket and then deletes the object version that was
    /// created.
    /// </summary>
    public class DeleteObjectVersion
    {
        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";
            string keyName = "verstioned-object.txt";

            // If the AWS Region of the default user is different from the AWS
            // Region of the Amazon S3 bucket, pass the AWS Region of the
            // bucket region to the Amazon S3 client object's constructor.
            // Define it like this:
            //      RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
            IAmazonS3 client = new AmazonS3Client();

            await CreateAndDeleteObjectVersionAsync(client, bucketName, keyName);
        }

        /// <summary>
        /// This method creates and then deletes a versioned object.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
        /// create and delete the object.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket where the
```

```
        /// object will be created and deleted.</param>
        /// <param name="keyName">The key name of the object to create.</param>
        public static async Task CreateAndDeleteObjectVersionAsync(IAmazonS3
 client, string bucketName, string keyName)
        {
            try
            {
                // Add a sample object.
                string versionID = await PutAnObject(client, bucketName,
 keyName);

                // Delete the object by specifying an object key and a version
 ID.
                DeleteObjectRequest request = new DeleteObjectRequest()
                {
                    BucketName = bucketName,
                    Key = keyName,
                    VersionId = versionID,
                };

                Console.WriteLine("Deleting an object");
                await client.DeleteObjectAsync(request);
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error: {ex.Message}");
            }
        }

        /// <summary>
        /// This method is used to create the temporary Amazon S3 object.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 object which will be
 used
        /// to create the temporary Amazon S3 object.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket where the
 object
        /// will be created.</param>
        /// <param name="objectKey">The name of the Amazon S3 object co create.</
param>
        /// <returns>The Version ID of the created object.</returns>
        public static async Task<string> PutAnObject(IAmazonS3 client, string
 bucketName, string objectKey)
        {
```

```
            PutObjectRequest request = new PutObjectRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
                ContentBody = "This is the content body!",
            };

            PutObjectResponse response = await client.PutObjectAsync(request);
            return response.VersionId;
        }
    }
```

- Untuk detail API, lihat DeleteObjectdi Referensi AWS SDK for .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function delete_item_in_bucket
#
# This function deletes the specified file from the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.
```

```
#       $2 - The key (file name) in the bucket to delete.

# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function delete_item_in_bucket() {
  local bucket_name=$1
  local key=$2
  local response

  response=$(aws s3api delete-object \
    --bucket "$bucket_name" \
    --key "$key")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api delete-object operation failed.\n
$response"
    return 1
  fi
}
```

- Untuk detail API, lihat DeleteObjectdi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::deleteObject(const Aws::String &objectKey,
                              const Aws::String &fromBucket,
                              const Aws::S3::S3ClientConfiguration &clientConfig)
 {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::DeleteObjectRequest request;
```

```
    request.WithKey(objectKey)
            .WithBucket(fromBucket);

    Aws::S3::Model::DeleteObjectOutcome outcome =
            client.DeleteObject(request);

    if (!outcome.IsSuccess()) {
        auto err = outcome.GetError();
        std::cerr << "Error: deleteObject: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    } else {
        std::cout << "Successfully deleted the object." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat DeleteObjectdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Perintah berikut menghapus objek bernama `test.txt` dari bucket bernama`my-bucket`:

```
aws s3api delete-object --bucket my-bucket --key test.txt
```

Jika pembuatan versi bucket diaktifkan, output akan berisi ID versi penanda hapus:

```
{
  "VersionId": "9_gKg5vG56F.TTEUdwkxGpJ3tNDlWlGq",
  "DeleteMarker": true
}
```

Untuk informasi selengkapnya tentang menghapus objek, lihat Menghapus Objek di Panduan Pengembang Amazon S3.

- Untuk detail API, lihat DeleteObjectdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// S3Actions wraps S3 service actions.
type S3Actions struct {
 S3Client  *s3.Client
 S3Manager *manager.Uploader
}



// DeleteObject deletes an object from a bucket.
func (actor S3Actions) DeleteObject(ctx context.Context, bucket string, key
 string, versionId string, bypassGovernance bool) (bool, error) {
 deleted := false
 input := &s3.DeleteObjectInput{
  Bucket: aws.String(bucket),
  Key:    aws.String(key),
 }
 if versionId != "" {
  input.VersionId = aws.String(versionId)
 }
 if bypassGovernance {
  input.BypassGovernanceRetention = aws.Bool(true)
 }
 _, err := actor.S3Client.DeleteObject(ctx, input)
 if err != nil {
  var noKey *types.NoSuchKey
  var apiErr *smithy.GenericAPIError
  if errors.As(err, &noKey) {
   log.Printf("Object %s does not exist in %s.\n", key, bucket)
   err = noKey
  } else if errors.As(err, &apiErr) {
   switch apiErr.ErrorCode() {
```

```
    case "AccessDenied":
     log.Printf("Access denied: cannot delete object %s from %s.\n", key, bucket)
     err = nil
    case "InvalidArgument":
     if bypassGovernance {
      log.Printf("You cannot specify bypass governance on a bucket without lock
 enabled.")
      err = nil
     }
    }
   }
  } else {
   deleted = true
  }
  return deleted, err
}
```

- Untuk detail API, lihat DeleteObjectdi Referensi AWS SDK for Go API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Hapus objek.

```
import { DeleteObjectCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new DeleteObjectCommand({
    Bucket: "test-bucket",
    Key: "test-key.txt",
  });
```

```
  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk detail API, lihat DeleteObjectdi Referensi AWS SDK for JavaScript API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Hapus objek.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    def delete(self):
        """
        Deletes the object.
        """
        try:
```

```
            self.object.delete()
            self.object.wait_until_not_exists()
            logger.info(
                "Deleted object '%s' from bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
        except ClientError:
            logger.exception(
                "Couldn't delete object '%s' from bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
            raise
```

Kembalikan objek ke versi sebelumnya dengan menghapus versi yang lebih baru dari objek tersebut.

```
def rollback_object(bucket, object_key, version_id):
    """
    Rolls back an object to an earlier version by deleting all versions that
    occurred after the specified rollback version.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that holds the object to roll back.
    :param object_key: The object to roll back.
    :param version_id: The version ID to roll back to.
    """
    # Versions must be sorted by last_modified date because delete markers are
    # at the end of the list even when they are interspersed in time.
    versions = sorted(
        bucket.object_versions.filter(Prefix=object_key),
        key=attrgetter("last_modified"),
        reverse=True,
    )

    logger.debug(
        "Got versions:\n%s",
        "\n".join(
```

```
            [
                f"\t{version.version_id}, last modified {version.last_modified}"
                for version in versions
            ]
        ),
    )

    if version_id in [ver.version_id for ver in versions]:
        print(f"Rolling back to version {version_id}")
        for version in versions:
            if version.version_id != version_id:
                version.delete()
                print(f"Deleted version {version.version_id}")
            else:
                break

        print(f"Active version is now {bucket.Object(object_key).version_id}")
    else:
        raise KeyError(
            f"{version_id} was not found in the list of versions for "
 f"{object_key}."
        )
```

Aktifkan kembali objek yang dihapus dengan menghapus penanda hapus aktif objek.

```
def revive_object(bucket, object_key):
    """
    Revives a versioned object that was deleted by removing the object's active
    delete marker.
    A versioned object presents as deleted when its latest version is a delete
 marker.
    By removing the delete marker, we make the previous version the latest
 version
    and the object then presents as *not* deleted.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that contains the object.
    :param object_key: The object to revive.
```

```
    """
    # Get the latest version for the object.
    response = s3.meta.client.list_object_versions(
        Bucket=bucket.name, Prefix=object_key, MaxKeys=1
    )

    if "DeleteMarkers" in response:
        latest_version = response["DeleteMarkers"][0]
        if latest_version["IsLatest"]:
            logger.info(
                "Object %s was indeed deleted on %s. Let's revive it.",
                object_key,
                latest_version["LastModified"],
            )
            obj = bucket.Object(object_key)
            obj.Version(latest_version["VersionId"]).delete()
            logger.info(
                "Revived %s, active version is now %s  with body '%s'",
                object_key,
                obj.version_id,
                obj.get()["Body"].read(),
            )
        else:
            logger.warning(
                "Delete marker is not the latest version for %s!", object_key
            )
    elif "Versions" in response:
        logger.warning("Got an active version for %s, nothing to do.",
 object_key)
    else:
        logger.error("Couldn't get any version info for %s.", object_key)
```

Buat penangan Lambda yang menghapus penanda hapus dari objek S3. Penangan ini dapat digunakan untuk membersihkan penanda hapus asing secara efisien dalam bucket berversi.

```
import logging
from urllib import parse
import boto3
from botocore.exceptions import ClientError
```

```
logger = logging.getLogger(__name__)
logger.setLevel("INFO")

s3 = boto3.client("s3")


def lambda_handler(event, context):
    """
    Removes a delete marker from the specified versioned object.

    :param event: The S3 batch event that contains the ID of the delete marker
                  to remove.
    :param context: Context about the event.
    :return: A result structure that Amazon S3 uses to interpret the result of
 the
             operation. When the result code is TemporaryFailure, S3 retries the
             operation.
    """
    # Parse job parameters from Amazon S3 batch operations
    invocation_id = event["invocationId"]
    invocation_schema_version = event["invocationSchemaVersion"]

    results = []
    result_code = None
    result_string = None

    task = event["tasks"][0]
    task_id = task["taskId"]

    try:
        obj_key = parse.unquote(task["s3Key"], encoding="utf-8")
        obj_version_id = task["s3VersionId"]
        bucket_name = task["s3BucketArn"].split(":")[-1]

        logger.info(
            "Got task: remove delete marker %s from object %s.", obj_version_id,
 obj_key
        )

        try:
            # If this call does not raise an error, the object version is not a
 delete
            # marker and should not be deleted.
            response = s3.head_object(
```

```
                Bucket=bucket_name, Key=obj_key, VersionId=obj_version_id
            )
            result_code = "PermanentFailure"
            result_string = (
                f"Object {obj_key}, ID {obj_version_id} is not " f"a delete
 marker."
            )

            logger.debug(response)
            logger.warning(result_string)
        except ClientError as error:
            delete_marker = error.response["ResponseMetadata"]
["HTTPHeaders"].get(
                "x-amz-delete-marker", "false"
            )
            if delete_marker == "true":
                logger.info(
                    "Object %s, version %s is a delete marker.", obj_key,
 obj_version_id
                )
                try:
                    s3.delete_object(
                        Bucket=bucket_name, Key=obj_key, VersionId=obj_version_id
                    )
                    result_code = "Succeeded"
                    result_string = (
                        f"Successfully removed delete marker "
                        f"{obj_version_id} from object {obj_key}."
                    )
                    logger.info(result_string)
                except ClientError as error:
                    # Mark request timeout as a temporary failure so it will be
 retried.
                    if error.response["Error"]["Code"] == "RequestTimeout":
                        result_code = "TemporaryFailure"
                        result_string = (
                            f"Attempt to remove delete marker from  "
                            f"object {obj_key} timed out."
                        )
                        logger.info(result_string)
                    else:
                        raise
            else:
                raise ValueError(
```

```
                    f"The x-amz-delete-marker header is either not "
                    f"present or is not 'true'."
                )
    except Exception as error:
        # Mark all other exceptions as permanent failures.
        result_code = "PermanentFailure"
        result_string = str(error)
        logger.exception(error)
    finally:
        results.append(
            {
                "taskId": task_id,
                "resultCode": result_code,
                "resultString": result_string,
            }
        )
    return {
        "invocationSchemaVersion": invocation_schema_version,
        "treatMissingKeysAs": "PermanentFailure",
        "invocationId": invocation_id,
        "results": results,
    }
```

- Untuk detail API, lihat DeleteObjectdi AWS SDK for Python (Boto3) Referensi API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
async fn remove_object(client: &Client, bucket: &str, key: &str) -> Result<(),
 Error> {
    client
        .delete_object()
```

```
        .bucket(bucket)
        .key(key)
        .send()
        .await?;

    println!("Object deleted.");

    Ok(())
}
```

- Untuk detail API, lihat DeleteObjectreferensi AWS SDK for Rust API.

SAP ABAP

SDK untuk SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    TRY.
        lo_s3->deleteobject(
            iv_bucket = iv_bucket_name
            iv_key = iv_object_key
        ).
        MESSAGE 'Object deleted from S3 bucket.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat DeleteObjectdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK untuk Swift

> **Note**
>
> Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
public func deleteFile(bucket: String, key: String) async throws {
    let input = DeleteObjectInput(
        bucket: bucket,
        key: key
    )

    do {
        _ = try await client.deleteObject(input: input)
    } catch {
        throw error
    }
}
```

- Untuk detail API, lihat referensi DeleteObject AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteObjectTagging`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteObjectTagging`.

CLI

AWS CLI

Untuk menghapus set tag dari suatu objek

`delete-object-tagging`Contoh berikut menghapus tag dengan kunci yang ditentukan dari objek`doc1.rtf`.

```
aws s3api delete-object-tagging \
    --bucket my-bucket \
    --key doc1.rtf
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat DeleteObjectTaggingdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menghapus semua tag yang terkait dengan objek dengan kunci 'testfile.txt' di Bucket S3 yang diberikan.

```
Remove-S3ObjectTagSet -Key 'testfile.txt' -BucketName 's3testbucket' -Select
 '^Key'
```

Output:

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-S3ObjectTagSet (DeleteObjectTagging)" on target
 "testfile.txt".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
testfile.txt
```

- Untuk detail API, lihat DeleteObjectTaggingdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteObjects`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteObjects`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai bucket dan objek

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Hapus semua objek dalam bucket S3.

```
        /// <summary>
        /// Delete all of the objects stored in an existing Amazon S3 bucket.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client object.</param>
        /// <param name="bucketName">The name of the bucket from which the
        /// contents will be deleted.</param>
        /// <returns>A boolean value that represents the success or failure of
        /// deleting all of the objects in the bucket.</returns>
        public static async Task<bool> DeleteBucketContentsAsync(IAmazonS3
client, string bucketName)
        {
            // Iterate over the contents of the bucket and delete all objects.
            var request = new ListObjectsV2Request
            {
                BucketName = bucketName,
```

```
            };

            try
            {
                ListObjectsV2Response response;

                do
                {
                    response = await client.ListObjectsV2Async(request);
                    response.S3Objects
                        .ForEach(async obj => await
client.DeleteObjectAsync(bucketName, obj.Key));

                    // If the response is truncated, set the request
ContinuationToken
                    // from the NextContinuationToken property of the response.
                    request.ContinuationToken = response.NextContinuationToken;
                }
                while (response.IsTruncated);

                return true;
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error deleting objects: {ex.Message}");
                return false;
            }
        }
```

Hapus beberapa objek dalam bucket S3 yang tidak berversi.

```
    using System;
    using System.Collections.Generic;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example shows how to delete multiple objects from an Amazon Simple
    /// Storage Service (Amazon S3) bucket.
    /// </summary>
```

```
   public class DeleteMultipleObjects
   {
       /// <summary>
       /// The Main method initializes the Amazon S3 client and the name of
       /// the bucket and then passes those values to MultiObjectDeleteAsync.
       /// </summary>
       public static async Task Main()
       {
           const string bucketName = "doc-example-bucket";

           // If the Amazon S3 bucket from which you wish to delete objects is
not
           // located in the same AWS Region as the default user, define the
           // AWS Region for the Amazon S3 bucket as a parameter to the client
           // constructor.
           IAmazonS3 s3Client = new AmazonS3Client();

           await MultiObjectDeleteAsync(s3Client, bucketName);
       }

       /// <summary>
       /// This method uses the passed Amazon S3 client to first create and then
       /// delete three files from the named bucket.
       /// </summary>
       /// <param name="client">The initialized Amazon S3 client object used to
call
       /// Amazon S3 methods.</param>
       /// <param name="bucketName">The name of the Amazon S3 bucket where
objects
       /// will be created and then deleted.</param>
       public static async Task MultiObjectDeleteAsync(IAmazonS3 client, string
bucketName)
       {
           // Create three sample objects which we will then delete.
           var keysAndVersions = await PutObjectsAsync(client, 3, bucketName);

           // Now perform the multi-object delete, passing the key names and
           // version IDs. Since we are working with a non-versioned bucket,
           // the object keys collection includes null version IDs.
           DeleteObjectsRequest multiObjectDeleteRequest = new
DeleteObjectsRequest
           {
               BucketName = bucketName,
               Objects = keysAndVersions,
```

```
            };

            // You can add a specific object key to the delete request using the
            // AddKey method of the multiObjectDeleteRequest.
            try
            {
                DeleteObjectsResponse response = await
client.DeleteObjectsAsync(multiObjectDeleteRequest);
                Console.WriteLine("Successfully deleted all the {0} items",
response.DeletedObjects.Count);
            }
            catch (DeleteObjectsException e)
            {
                PrintDeletionErrorStatus(e);
            }
        }

        /// <summary>
        /// Prints the list of errors raised by the call to DeleteObjectsAsync.
        /// </summary>
        /// <param name="ex">A collection of exceptions returned by the call to
        /// DeleteObjectsAsync.</param>
        public static void PrintDeletionErrorStatus(DeleteObjectsException ex)
        {
            DeleteObjectsResponse errorResponse = ex.Response;
            Console.WriteLine("x {0}", errorResponse.DeletedObjects.Count);

            Console.WriteLine($"Successfully deleted
{errorResponse.DeletedObjects.Count}.");
            Console.WriteLine($"No. of objects failed to delete =
{errorResponse.DeleteErrors.Count}");

            Console.WriteLine("Printing error data...");
            foreach (DeleteError deleteError in errorResponse.DeleteErrors)
            {
                Console.WriteLine($"Object Key:
{deleteError.Key}\t{deleteError.Code}\t{deleteError.Message}");
            }
        }

        /// <summary>
        /// This method creates simple text file objects that can be used in
        /// the delete method.
        /// </summary>
```

```
        /// <param name="client">The Amazon S3 client used to call
PutObjectAsync.</param>
        /// <param name="number">The number of objects to create.</param>
        /// <param name="bucketName">The name of the bucket where the objects
        /// will be created.</param>
        /// <returns>A list of keys (object keys) and versions that the calling
        /// method will use to delete the newly created files.</returns>
        public static async Task<List<KeyVersion>> PutObjectsAsync(IAmazonS3
client, int number, string bucketName)
        {
            List<KeyVersion> keys = new List<KeyVersion>();
            for (int i = 0; i < number; i++)
            {
                string key = "ExampleObject-" + new System.Random().Next();
                PutObjectRequest request = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = key,
                    ContentBody = "This is the content body!",
                };

                PutObjectResponse response = await
client.PutObjectAsync(request);

                // For non-versioned bucket operations, we only need the
                // object key.
                KeyVersion keyVersion = new KeyVersion
                {
                    Key = key,
                };
                keys.Add(keyVersion);
            }

            return keys;
        }
    }
```

Hapus beberapa objek dalam bucket S3 berversi.

```
    using System;
    using System.Collections.Generic;
```

```
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example shows how to delete objects in a version-enabled Amazon
    /// Simple StorageService (Amazon S3) bucket.
    /// </summary>
    public class DeleteMultipleObjects
    {
        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";

            // If the AWS Region for your Amazon S3 bucket is different from
            // the AWS Region of the default user, define the AWS Region for
            // the Amazon S3 bucket and pass it to the client constructor
            // like this:
            // RegionEndpoint bucketRegion = RegionEndpoint.USWest2;
            IAmazonS3 s3Client;

            s3Client = new AmazonS3Client();
            await DeleteMultipleObjectsFromVersionedBucketAsync(s3Client,
bucketName);
        }

        /// <summary>
        /// This method removes multiple versions and objects from a
        /// version-enabled Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// DeleteObjectVersionsAsync, DeleteObjectsAsync, and
        /// RemoveDeleteMarkersAsync.</param>
        /// <param name="bucketName">The name of the bucket from which to delete
        /// objects.</param>
        public static async Task
DeleteMultipleObjectsFromVersionedBucketAsync(IAmazonS3 client, string
bucketName)
        {
            // Delete objects (specifying object version in the request).
            await DeleteObjectVersionsAsync(client, bucketName);

            // Delete objects (without specifying object version in the request).
```

```
            var deletedObjects = await DeleteObjectsAsync(client, bucketName);

            // Additional exercise - remove the delete markers Amazon S3 returned
from
            // the preceding response. This results in the objects reappearing
            // in the bucket (you can verify the appearance/disappearance of
            // objects in the console).
            await RemoveDeleteMarkersAsync(client, bucketName, deletedObjects);
        }

        /// <summary>
        /// Creates and then deletes non-versioned Amazon S3 objects and then
deletes
        /// them again. The method returns a list of the Amazon S3 objects
deleted.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// PubObjectsAsync and NonVersionedDeleteAsync.</param>
        /// <param name="bucketName">The name of the bucket where the objects
        /// will be created and then deleted.</param>
        /// <returns>A list of DeletedObjects.</returns>
        public static async Task<List<DeletedObject>>
DeleteObjectsAsync(IAmazonS3 client, string bucketName)
        {
            // Upload the sample objects.
            var keysAndVersions2 = await PutObjectsAsync(client, bucketName, 3);

            // Delete objects using only keys. Amazon S3 creates a delete marker
and
            // returns its version ID in the response.
            List<DeletedObject> deletedObjects = await
NonVersionedDeleteAsync(client, bucketName, keysAndVersions2);
            return deletedObjects;
        }

        /// <summary>
        /// This method creates several temporary objects and then deletes them.
        /// </summary>
        /// <param name="client">The S3 client.</param>
        /// <param name="bucketName">Name of the bucket.</param>
        /// <returns>Async task.</returns>
        public static async Task DeleteObjectVersionsAsync(IAmazonS3 client,
string bucketName)
```

```
        {
            // Upload the sample objects.
            var keysAndVersions1 = await PutObjectsAsync(client, bucketName, 3);

            // Delete the specific object versions.
            await VersionedDeleteAsync(client, bucketName, keysAndVersions1);
        }

        /// <summary>
        /// Displays the list of information about deleted files to the console.
        /// </summary>
        /// <param name="e">Error information from the delete process.</param>
        private static void DisplayDeletionErrors(DeleteObjectsException e)
        {
            var errorResponse = e.Response;
            Console.WriteLine($"No. of objects successfully deleted =
 {errorResponse.DeletedObjects.Count}");
            Console.WriteLine($"No. of objects failed to delete =
 {errorResponse.DeleteErrors.Count}");
            Console.WriteLine("Printing error data...");
            foreach (var deleteError in errorResponse.DeleteErrors)
            {
                Console.WriteLine($"Object Key:
 {deleteError.Key}\t{deleteError.Code}\t{deleteError.Message}");
            }
        }

        /// <summary>
        /// Delete multiple objects from a version-enabled bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
 call
        /// DeleteObjectVersionsAsync, DeleteObjectsAsync, and
        /// RemoveDeleteMarkersAsync.</param>
        /// <param name="bucketName">The name of the bucket from which to delete
        /// objects.</param>
        /// <param name="keys">A list of key names for the objects to delete.</
param>
        private static async Task VersionedDeleteAsync(IAmazonS3 client, string
 bucketName, List<KeyVersion> keys)
        {
            var multiObjectDeleteRequest = new DeleteObjectsRequest
            {
                BucketName = bucketName,
```

```
                Objects = keys, // This includes the object keys and specific
 version IDs.
            };

            try
            {
                Console.WriteLine("Executing VersionedDelete...");
                DeleteObjectsResponse response = await
 client.DeleteObjectsAsync(multiObjectDeleteRequest);
                Console.WriteLine($"Successfully deleted all the
 {response.DeletedObjects.Count} items");
            }
            catch (DeleteObjectsException ex)
            {
                DisplayDeletionErrors(ex);
            }
        }

        /// <summary>
        /// Deletes multiple objects from a non-versioned Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
 call
        /// DeleteObjectVersionsAsync, DeleteObjectsAsync, and
        /// RemoveDeleteMarkersAsync.</param>
        /// <param name="bucketName">The name of the bucket from which to delete
        /// objects.</param>
        /// <param name="keys">A list of key names for the objects to delete.</
param>
        /// <returns>A list of the deleted objects.</returns>
        private static async Task<List<DeletedObject>>
 NonVersionedDeleteAsync(IAmazonS3 client, string bucketName, List<KeyVersion>
 keys)
        {
            // Create a request that includes only the object key names.
            DeleteObjectsRequest multiObjectDeleteRequest = new
 DeleteObjectsRequest();
            multiObjectDeleteRequest.BucketName = bucketName;

            foreach (var key in keys)
            {
                multiObjectDeleteRequest.AddKey(key.Key);
            }
```

```
            // Execute DeleteObjectsAsync.
            // The DeleteObjectsAsync method adds a delete marker for each
            // object deleted. You can verify that the objects were removed
            // using the Amazon S3 console.
            DeleteObjectsResponse response;
            try
            {
                Console.WriteLine("Executing NonVersionedDelete...");
                response = await
 client.DeleteObjectsAsync(multiObjectDeleteRequest);
                Console.WriteLine("Successfully deleted all the {0} items",
 response.DeletedObjects.Count);
            }
            catch (DeleteObjectsException ex)
            {
                DisplayDeletionErrors(ex);
                throw; // Some deletions failed. Investigate before continuing.
            }

            // This response contains the DeletedObjects list which we use to
 delete the delete markers.
            return response.DeletedObjects;
        }

        /// <summary>
        /// Deletes the markers left after deleting the temporary objects.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
 call
        /// DeleteObjectVersionsAsync, DeleteObjectsAsync, and
        /// RemoveDeleteMarkersAsync.</param>
        /// <param name="bucketName">The name of the bucket from which to delete
        /// objects.</param>
        /// <param name="deletedObjects">A list of the objects that were
 deleted.</param>
        private static async Task RemoveDeleteMarkersAsync(IAmazonS3 client,
 string bucketName, List<DeletedObject> deletedObjects)
        {
            var keyVersionList = new List<KeyVersion>();

            foreach (var deletedObject in deletedObjects)
            {
                KeyVersion keyVersion = new KeyVersion
                {
```

```
                    Key = deletedObject.Key,
                    VersionId = deletedObject.DeleteMarkerVersionId,
                };
                keyVersionList.Add(keyVersion);
            }

            // Create another request to delete the delete markers.
            var multiObjectDeleteRequest = new DeleteObjectsRequest
            {
                BucketName = bucketName,
                Objects = keyVersionList,
            };

            // Now, delete the delete marker to bring your objects back to the
 bucket.
            try
            {
                Console.WriteLine("Removing the delete markers .....");
                var deleteObjectResponse = await
 client.DeleteObjectsAsync(multiObjectDeleteRequest);
                Console.WriteLine($"Successfully deleted the
 {deleteObjectResponse.DeletedObjects.Count} delete markers");
            }
            catch (DeleteObjectsException ex)
            {
                DisplayDeletionErrors(ex);
            }
        }

        /// <summary>
        /// Create temporary Amazon S3 objects to show how object deletion wors
 in an
        /// Amazon S3 bucket with versioning enabled.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
 call
        /// PutObjectAsync to create temporary objects for the example.</param>
        /// <param name="bucketName">A string representing the name of the S3
        /// bucket where we will create the temporary objects.</param>
        /// <param name="number">The number of temporary objects to create.</
param>
        /// <returns>A list of the KeyVersion objects.</returns>
        private static async Task<List<KeyVersion>> PutObjectsAsync(IAmazonS3
 client, string bucketName, int number)
```

```
        {
            var keys = new List<KeyVersion>();

            for (var i = 0; i < number; i++)
            {
                string key = "ObjectToDelete-" + new System.Random().Next();
                PutObjectRequest request = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = key,
                    ContentBody = "This is the content body!",
                };

                var response = await client.PutObjectAsync(request);
                KeyVersion keyVersion = new KeyVersion
                {
                    Key = key,
                    VersionId = response.VersionId,
                };

                keys.Add(keyVersion);
            }

            return keys;
        }
    }
```

- Untuk detail API, lihat DeleteObjectsdi Referensi AWS SDK for .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
###########################################################################
```

```
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function delete_items_in_bucket
#
# This function deletes the specified list of keys from the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.
#       $2 - A list of keys in the bucket to delete.

# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function delete_items_in_bucket() {
  local bucket_name=$1
  local keys=$2
  local response

  # Create the JSON for the items to delete.
  local delete_items
  delete_items="{\"Objects\":["
  for key in $keys; do
    delete_items="$delete_items{\"Key\": \"$key\"},"
  done
  delete_items=${delete_items%?} # Remove the final comma.
  delete_items="$delete_items]}"

  response=$(aws s3api delete-objects \
    --bucket "$bucket_name" \
    --delete "$delete_items")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api delete-object operation failed.\n
$response"
    return 1
```

```
  fi
}
```

- Untuk detail API, lihat DeleteObjectsdi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::deleteObjects(const std::vector<Aws::String> &objectKeys,
                               const Aws::String &fromBucket,
                               const Aws::S3::S3ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::DeleteObjectsRequest request;

    Aws::S3::Model::Delete deleteObject;
    for (const Aws::String &objectKey: objectKeys) {

 deleteObject.AddObjects(Aws::S3::Model::ObjectIdentifier().WithKey(objectKey));
    }

    request.SetDelete(deleteObject);
    request.SetBucket(fromBucket);

    Aws::S3::Model::DeleteObjectsOutcome outcome =
            client.DeleteObjects(request);

    if (!outcome.IsSuccess()) {
        auto err = outcome.GetError();
        std::cerr << "Error deleting objects. " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    } else {
        std::cout << "Successfully deleted the objects.";
```

```
        for (size_t i = 0; i < objectKeys.size(); ++i) {
            std::cout << objectKeys[i];
            if (i < objectKeys.size() - 1) {
                std::cout << ", ";
            }
        }

        std::cout << " from bucket " << fromBucket << "." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat DeleteObjectsdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Perintah berikut menghapus objek dari bucket bernama`my-bucket`:

```
aws s3api delete-objects --bucket my-bucket --delete file://delete.json
```

`delete.json`adalah dokumen JSON di direktori saat ini yang menentukan objek yang akan dihapus:

```
{
  "Objects": [
    {
      "Key": "test1.txt"
    }
  ],
  "Quiet": false
}
```

Output:

```
{
    "Deleted": [
        {
            "DeleteMarkerVersionId": "mYAT5Mc6F7aeUL8SS7FAAqUPO1koHwzU",
```

```
            "Key": "test1.txt",
            "DeleteMarker": true
        }
    ]
}
```

- Untuk detail API, lihat DeleteObjectsdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// S3Actions wraps S3 service actions.
type S3Actions struct {
 S3Client  *s3.Client
 S3Manager *manager.Uploader
}



// DeleteObjects deletes a list of objects from a bucket.
func (actor S3Actions) DeleteObjects(ctx context.Context, bucket string, objects
 []types.ObjectIdentifier, bypassGovernance bool) error {
 if len(objects) == 0 {
  return nil
 }

 input := s3.DeleteObjectsInput{
  Bucket: aws.String(bucket),
  Delete: &types.Delete{
   Objects: objects,
   Quiet:   aws.Bool(true),
  },
 }
 if bypassGovernance {
```

```
  input.BypassGovernanceRetention = aws.Bool(true)
 }
 delOut, err := actor.S3Client.DeleteObjects(ctx, &input)
 if err != nil || len(delOut.Errors) > 0 {
  log.Printf("Error deleting objects from bucket %s.\n", bucket)
  if err != nil {
   var noBucket *types.NoSuchBucket
   if errors.As(err, &noBucket) {
    log.Printf("Bucket %s does not exist.\n", bucket)
    err = noBucket
   }
  } else if len(delOut.Errors) > 0 {
   for _, outErr := range delOut.Errors {
    log.Printf("%s: %s\n", *outErr.Key, *outErr.Message)
   }
   err = fmt.Errorf("%s", *delOut.Errors[0].Message)
  }
 }
 return err
}
```

- Untuk detail API, lihat DeleteObjectsdi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.model.ObjectIdentifier;
import software.amazon.awssdk.services.s3.model.Delete;
import software.amazon.awssdk.services.s3.model.DeleteObjectsRequest;
```

```
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.util.ArrayList;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class DeleteMultiObjects {
    public static void main(String[] args) {
        final String usage = """

                Usage:    <bucketName>

                Where:
                   bucketName - the Amazon S3 bucket name.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        deleteBucketObjects(s3, bucketName);
        s3.close();
    }

    public static void deleteBucketObjects(S3Client s3, String bucketName) {
        // Upload three sample objects to the specfied Amazon S3 bucket.
        ArrayList<ObjectIdentifier> keys = new ArrayList<>();
        PutObjectRequest putOb;
        ObjectIdentifier objectId;
```

```
        for (int i = 0; i < 3; i++) {
            String keyName = "delete object example " + i;
            objectId = ObjectIdentifier.builder()
                    .key(keyName)
                    .build();

            putOb = PutObjectRequest.builder()
                    .bucket(bucketName)
                    .key(keyName)
                    .build();

            s3.putObject(putOb, RequestBody.fromString(keyName));
            keys.add(objectId);
        }

        System.out.println(keys.size() + " objects successfully created.");

        // Delete multiple objects in one request.
        Delete del = Delete.builder()
                .objects(keys)
                .build();

        try {
            DeleteObjectsRequest multiObjectDeleteRequest =
 DeleteObjectsRequest.builder()
                    .bucket(bucketName)
                    .delete(del)
                    .build();

            s3.deleteObjects(multiObjectDeleteRequest);
            System.out.println("Multiple objects are deleted!");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat DeleteObjectsdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Hapus beberapa objek.

```
import { DeleteObjectsCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new DeleteObjectsCommand({
    Bucket: "test-bucket",
    Delete: {
      Objects: [{ Key: "object1.txt" }, { Key: "object2.txt" }],
    },
  });

  try {
    const { Deleted } = await client.send(command);
    console.log(
      `Successfully deleted ${Deleted.length} objects from S3 bucket. Deleted
 objects:`,
    );
    console.log(Deleted.map((d) => ` • ${d.Key}`).join("\n"));
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk detail API, lihat DeleteObjectsdi Referensi AWS SDK for JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun deleteBucketObjects(
    bucketName: String,
    objectName: String,
) {
    val objectId =
        ObjectIdentifier {
            key = objectName
        }

    val delOb =
        Delete {
            objects = listOf(objectId)
        }

    val request =
        DeleteObjectsRequest {
            bucket = bucketName
            delete = delOb
        }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.deleteObjects(request)
        println("$objectName was deleted from $bucketName")
    }
}
```

- Untuk detail API, lihat DeleteObjectsdi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Hapus satu set objek dari daftar kunci.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        try {
            $objects = [];
            foreach ($contents['Contents'] as $content) {
                $objects[] = [
                    'Key' => $content['Key'],
                ];
            }
            $this->s3client->deleteObjects([
                'Bucket' => $this->bucketName,
                'Delete' => [
                    'Objects' => $objects,
                ],
            ]);
            $check = $this->s3client->listObjectsV2([
                'Bucket' => $this->bucketName,
            ]);
            if (count($check) <= 0) {
                throw new Exception("Bucket wasn't empty.");
            }
            echo "Deleted all objects and folders from $this->bucketName.\n";
        } catch (Exception $exception) {
            echo "Failed to delete $fileName from $this->bucketName with error:
" . $exception->getMessage();
            exit("Please fix error with object deletion before continuing.");
        }
```

- Untuk detail API, lihat DeleteObjectsdi Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menghapus objek "sample.txt" dari bucket “test-files”. Anda diminta konfirmasi sebelum perintah dijalankan; untuk menekan prompt gunakan sakelar -Force.

```
Remove-S3Object -BucketName test-files -Key sample.txt
```

Contoh 2: Perintah ini menghapus versi objek "sample.txt" yang ditentukan dari bucket “test-files”, dengan asumsi bucket telah dikonfigurasi untuk mengaktifkan versi objek.

```
Remove-S3Object -BucketName test-files -Key sample.txt -VersionId
 HLbxnx6V9omT6AQYVpks8mmFKQcejpqt
```

Contoh 3: Perintah ini menghapus objek "sample1.txt “," sample2.txt "dan" sample3.txt "dari bucket “test-files” sebagai operasi batch tunggal. Respons layanan akan mencantumkan semua kunci yang diproses, terlepas dari status keberhasilan atau kesalahan penghapusan. Untuk mendapatkan hanya kesalahan untuk kunci yang tidak dapat diproses oleh layanan tambahkan ReportErrorsOnly parameter - (parameter ini juga dapat ditentukan dengan alias -Quiet.

```
Remove-S3Object -BucketName test-files -KeyCollection @( "sample1.txt",
 "sample2.txt", "sample3.txt" )
```

Contoh 4: Contoh ini menggunakan ekspresi sebaris dengan KeyCollection parameter - untuk mendapatkan kunci objek yang akan dihapus. Get-S3Objectmengembalikan koleksi contoh Amazon.S3.Model.S3Object, yang masing-masing memiliki anggota Key dari jenis string mengidentifikasi objek.

```
Remove-S3Object -bucketname "test-files" -KeyCollection (Get-S3Object "test-
files" -KeyPrefix "prefix/subprefix" | select -ExpandProperty Key)
```

Contoh 5: Contoh ini memperoleh semua objek yang memiliki key prefix “prefix/subprefix” di bucket dan menghapusnya. Perhatikan bahwa objek yang masuk diproses satu per satu. Untuk koleksi besar, pertimbangkan untuk meneruskan koleksi ke parameter cmdlet's -InputObject (alias -S3ObjectCollection) untuk memungkinkan penghapusan terjadi sebagai batch dengan satu panggilan ke layanan.

```
Get-S3Object -BucketName "test-files" -KeyPrefix "prefix/subprefix" | Remove-
S3Object -Force
```

Contoh 6: Contoh ini menyalurkan kumpulan ObjectVersion instance Amazon.S3.Model.S3 yang mewakili penanda hapus ke cmdlet untuk dihapus. Perhatikan bahwa objek yang masuk diproses satu per satu. Untuk koleksi besar, pertimbangkan untuk meneruskan koleksi ke parameter cmdlet's - InputObject (alias -S3ObjectCollection) untuk memungkinkan penghapusan terjadi sebagai batch dengan satu panggilan ke layanan.

```
(Get-S3Version -BucketName "test-files").Versions | Where {$_.IsDeleteMarker -eq
 "True"} | Remove-S3Object -Force
```

Contoh 7: Script ini menunjukkan bagaimana melakukan penghapusan batch dari satu set objek (dalam hal ini menghapus penanda) dengan membangun array objek yang akan digunakan dengan - KeyAndVersionCollection parameter.

```
$keyVersions = @()
$markers = (Get-S3Version -BucketName $BucketName).Versions | Where
 {$_.IsDeleteMarker -eq "True"}
foreach ($marker in $markers) { $keyVersions += @{ Key = $marker.Key; VersionId =
 $marker.VersionId } }
Remove-S3Object -BucketName $BucketName -KeyAndVersionCollection $keyVersions -
Force
```

- Untuk detail API, lihat DeleteObjectsdi Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Hapus satu set objek dengan menggunakan daftar kunci objek.

```
class ObjectWrapper:
```

```
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    @staticmethod
    def delete_objects(bucket, object_keys):
        """
        Removes a list of objects from a bucket.
        This operation is done as a batch in a single request.

        :param bucket: The bucket that contains the objects. This is a Boto3
Bucket
                       resource.
        :param object_keys: The list of keys that identify the objects to remove.
        :return: The response that contains data about which objects were deleted
                 and any that could not be deleted.
        """
        try:
            response = bucket.delete_objects(
                Delete={"Objects": [{"Key": key} for key in object_keys]}
            )
            if "Deleted" in response:
                logger.info(
                    "Deleted objects '%s' from bucket '%s'.",
                    [del_obj["Key"] for del_obj in response["Deleted"]],
                    bucket.name,
                )
            if "Errors" in response:
                logger.warning(
                    "Could not delete objects '%s' from bucket '%s'.",
                    [
                        f"{del_obj['Key']}: {del_obj['Code']}"
                        for del_obj in response["Errors"]
                    ],
                    bucket.name,
                )
```

```
        except ClientError:
            logger.exception("Couldn't delete any objects from bucket %s.",
 bucket.name)
            raise
        else:
            return response
```

Hapus semua objek dalam bucket.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    @staticmethod
    def empty_bucket(bucket):
        """
        Remove all objects from a bucket.

        :param bucket: The bucket to empty. This is a Boto3 Bucket resource.
        """
        try:
            bucket.objects.delete()
            logger.info("Emptied bucket '%s'.", bucket.name)
        except ClientError:
            logger.exception("Couldn't empty bucket '%s'.", bucket.name)
            raise
```

Hapus objek berversi secara permanen dengan menghapus semua versinya.

```
def permanently_delete_object(bucket, object_key):
```

```
    """
    Permanently deletes a versioned object by deleting all of its versions.

    Usage is shown in the usage_demo_single_object function at the end of this
module.

    :param bucket: The bucket that contains the object.
    :param object_key: The object to delete.
    """
    try:
        bucket.object_versions.filter(Prefix=object_key).delete()
        logger.info("Permanently deleted all versions of object %s.", object_key)
    except ClientError:
        logger.exception("Couldn't delete all versions of %s.", object_key)
        raise
```

- Untuk detail API, lihat DeleteObjectsdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
 # Deletes the objects in an Amazon S3 bucket and deletes the bucket.
 #
 # @param bucket [Aws::S3::Bucket] The bucket to empty and delete.
 def delete_bucket(bucket)
   puts("\nDo you want to delete all of the objects as well as the bucket (y/n)?
")
   answer = gets.chomp.downcase
   if answer == "y"
     bucket.objects.batch_delete!
     bucket.delete
     puts("Emptied and deleted bucket #{bucket.name}.\n")
```

```
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't empty and delete bucket #{bucket.name}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  end
```

- Untuk detail API, lihat DeleteObjectsdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
pub async fn delete_objects(client: &Client, bucket_name: &str) ->
 Result<Vec<String>, Error> {
    let objects = client.list_objects_v2().bucket(bucket_name).send().await?;

    let mut delete_objects: Vec<ObjectIdentifier> = vec![];
    for obj in objects.contents() {
        let obj_id = ObjectIdentifier::builder()
            .set_key(Some(obj.key().unwrap().to_string()))
            .build()
            .map_err(Error::from)?;
        delete_objects.push(obj_id);
    }

    let return_keys = delete_objects.iter().map(|o| o.key.clone()).collect();

    if !delete_objects.is_empty() {
        client
            .delete_objects()
            .bucket(bucket_name)
            .delete(
                Delete::builder()
                    .set_objects(Some(delete_objects))
```

```
                    .build()
                    .map_err(Error::from)?,
            )
            .send()
            .await?;
    }

    let objects: ListObjectsV2Output =
 client.list_objects_v2().bucket(bucket_name).send().await?;

    eprintln!("{objects:?}");

    match objects.key_count {
        Some(0) => Ok(return_keys),
        _ => Err(Error::unhandled(
            "There were still objects left in the bucket.",
        )),
    }
}
```

- Untuk detail API, lihat DeleteObjectsreferensi AWS SDK for Rust API.

Swift

SDK untuk Swift

> **Note**
>
> Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    public func deleteObjects(bucket: String, keys: [String]) async throws {
```

```
        let input = DeleteObjectsInput(
            bucket: bucket,
            delete: S3ClientTypes.Delete(
                objects: keys.map({ S3ClientTypes.ObjectIdentifier(key: $0) }),
                quiet: true
            )
        )

        do {
            let output = try await client.deleteObjects(input: input)

            // As of the last update to this example, any errors are returned
            // in the `output` object's `errors` property. If there are any
            // errors in this array, throw an exception. Once the error
            // handling is finalized in later updates to the AWS SDK for
            // Swift, this example will be updated to handle errors better.

            guard let errors = output.errors else {
                return  // No errors.
            }
            if errors.count != 0 {
                throw ServiceHandlerError.deleteObjectsError
            }
        } catch {
            throw error
        }
    }
```

- Untuk detail API, lihat referensi DeleteObjects AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeletePublicAccessBlock`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeletePublicAccessBlock`.

CLI

AWS CLI

Untuk menghapus konfigurasi blokir akses publik untuk bucket

`delete-public-access-block`Contoh berikut menghapus konfigurasi blokir akses publik pada bucket yang ditentukan.

```
aws s3api delete-public-access-block \
    --bucket my-bucket
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat DeletePublicAccessBlockdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mematikan setelan blokir akses publik untuk bucket yang diberikan.

```
Remove-S3PublicAccessBlock -BucketName 's3testbucket' -Force -Select
 '^BucketName'
```

Output:

```
s3testbucket
```

- Untuk detail API, lihat DeletePublicAccessBlockdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketAccelerateConfiguration`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketAccelerateConfiguration`.

CLI

AWS CLI

Untuk mengambil konfigurasi percepatan bucket

`get-bucket-accelerate-configuration`Contoh berikut mengambil konfigurasi percepatan untuk bucket yang ditentukan.

```
aws s3api get-bucket-accelerate-configuration \
    --bucket my-bucket
```

Output:

```
{
    "Status": "Enabled"
}
```

- Untuk detail API, lihat GetBucketAccelerateConfigurationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan nilai Diaktifkan, jika pengaturan akselerasi transfer diaktifkan untuk bucket yang ditentukan.

```
Get-S3BucketAccelerateConfiguration -BucketName 's3testbucket'
```

Output:

```
Value
-----
Enabled
```

- Untuk detail API, lihat GetBucketAccelerateConfigurationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketAcl`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketAcl`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Mengelola daftar kontrol akses (ACL)

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        /// <summary>
        /// Get the access control list (ACL) for the new bucket.
        /// </summary>
        /// <param name="client">The initialized client object used to get the
        /// access control list (ACL) of the bucket.</param>
        /// <param name="newBucketName">The name of the newly created bucket.</
param>
        /// <returns>An S3AccessControlList.</returns>
        public static async Task<S3AccessControlList>
 GetACLForBucketAsync(IAmazonS3 client, string newBucketName)
        {
            // Retrieve bucket ACL to show that the ACL was properly applied to
            // the new bucket.
            GetACLResponse getACLResponse = await client.GetACLAsync(new
 GetACLRequest
            {
                BucketName = newBucketName,
```

```
            });

            return getACLResponse.AccessControlList;
        }
```

- Untuk detail API, lihat GetBucketAcldi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::getBucketAcl(const Aws::String &bucketName,
                              const Aws::S3::S3ClientConfiguration &clientConfig)
 {
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::GetBucketAclRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::GetBucketAclOutcome outcome =
            s3Client.GetBucketAcl(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: getBucketAcl: "
                  << err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    } else {
        Aws::Vector<Aws::S3::Model::Grant> grants =
                outcome.GetResult().GetGrants();

        for (auto it = grants.begin(); it != grants.end(); it++) {
            Aws::S3::Model::Grant grant = *it;
            Aws::S3::Model::Grantee grantee = grant.GetGrantee();
```

```
            std::cout << "For bucket " << bucketName << ": "
                      << std::endl << std::endl;

            if (grantee.TypeHasBeenSet()) {
                std::cout << "Type:          "
                          << getGranteeTypeString(grantee.GetType()) <<
 std::endl;
            }

            if (grantee.DisplayNameHasBeenSet()) {
                std::cout << "Display name:  "
                          << grantee.GetDisplayName() << std::endl;
            }

            if (grantee.EmailAddressHasBeenSet()) {
                std::cout << "Email address: "
                          << grantee.GetEmailAddress() << std::endl;
            }

            if (grantee.IDHasBeenSet()) {
                std::cout << "ID:            "
                          << grantee.GetID() << std::endl;
            }

            if (grantee.URIHasBeenSet()) {
                std::cout << "URI:           "
                          << grantee.GetURI() << std::endl;
            }

            std::cout << "Permission:    " <<
                      getPermissionString(grant.GetPermission()) <<
                      std::endl << std::endl;
        }
    }

    return outcome.IsSuccess();
}

//! Routine which converts a built-in type enumeration to a human-readable
 string.
/*!
 \param type: Type enumeration.
 \return String: Human-readable string.
```

```
*/

Aws::String getGranteeTypeString(const Aws::S3::Model::Type &type) {
    switch (type) {
        case Aws::S3::Model::Type::AmazonCustomerByEmail:
            return "Email address of an AWS account";
        case Aws::S3::Model::Type::CanonicalUser:
            return "Canonical user ID of an AWS account";
        case Aws::S3::Model::Type::Group:
            return "Predefined Amazon S3 group";
        case Aws::S3::Model::Type::NOT_SET:
            return "Not set";
        default:
            return "Type unknown";
    }
}

//! Routine which converts a built-in type enumeration to a human-readable
 string.
/*!
 \param permission: Permission enumeration.
 \return String: Human-readable string.
*/

Aws::String getPermissionString(const Aws::S3::Model::Permission &permission) {
    switch (permission) {
        case Aws::S3::Model::Permission::FULL_CONTROL:
            return "Can list objects in this bucket, create/overwrite/delete "
                   "objects in this bucket, and read/write this "
                   "bucket's permissions";
        case Aws::S3::Model::Permission::NOT_SET:
            return "Permission not set";
        case Aws::S3::Model::Permission::READ:
            return "Can list objects in this bucket";
        case Aws::S3::Model::Permission::READ_ACP:
            return "Can read this bucket's permissions";
        case Aws::S3::Model::Permission::WRITE:
            return "Can create, overwrite, and delete objects in this bucket";
        case Aws::S3::Model::Permission::WRITE_ACP:
            return "Can write this bucket's permissions";
        default:
            return "Permission unknown";
    }
```

```
    return "Permission unknown";
}
```

- Untuk detail API, lihat GetBucketAcldi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Perintah berikut mengambil daftar kontrol akses untuk bucket bernamamy-bucket:

```
aws s3api get-bucket-acl --bucket my-bucket
```

Output:

```
{
    "Owner": {
        "DisplayName": "my-username",
        "ID": "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "my-username",
                "ID":
 "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
            },
            "Permission": "FULL_CONTROL"
        }
    ]
}
```

- Untuk detail API, lihat GetBucketAcldi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectAclRequest;
import software.amazon.awssdk.services.s3.model.GetObjectAclResponse;
import software.amazon.awssdk.services.s3.model.Grant;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class GetAcl {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                  <bucketName> <objectKey>

                Where:
                  bucketName - The Amazon S3 bucket to get the access control
 list (ACL) for.
                  objectKey - The object to get the ACL for.\s
                """;

        if (args.length != 2) {
```

```
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String objectKey = args[1];
        System.out.println("Retrieving ACL for object: " + objectKey);
        System.out.println("in bucket: " + bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        getBucketACL(s3, objectKey, bucketName);
        s3.close();
        System.out.println("Done!");
    }

    public static String getBucketACL(S3Client s3, String objectKey, String
 bucketName) {
        try {
            GetObjectAclRequest aclReq = GetObjectAclRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .build();

            GetObjectAclResponse aclRes = s3.getObjectAcl(aclReq);
            List<Grant> grants = aclRes.grants();
            String grantee = "";
            for (Grant grant : grants) {
                System.out.format("  %s: %s\n", grant.grantee().id(),
 grant.permission());
                grantee = grant.grantee().id();
            }

            return grantee;
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

        return "";
    }
}
```

- Untuk detail API, lihat GetBucketAcldi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Dapatkan izin ACL.

```
import { GetBucketAclCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new GetBucketAclCommand({
    Bucket: "test-bucket",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat GetBucketAcldi Referensi AWS SDK for JavaScript API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def get_acl(self):
        """
        Get the ACL of the bucket.

        :return: The ACL of the bucket.
        """
        try:
            acl = self.bucket.Acl()
            logger.info(
                "Got ACL for bucket %s. Owner is %s.", self.bucket.name,
 acl.owner
            )
        except ClientError:
            logger.exception("Couldn't get ACL for bucket %s.", self.bucket.name)
            raise
        else:
            return acl
```

- Untuk detail API, lihat GetBucketAcldi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketAnalyticsConfiguration`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketAnalyticsConfiguration`.

CLI

AWS CLI

Untuk mengambil konfigurasi analitik untuk bucket dengan ID tertentu

`get-bucket-analytics-configuration`Contoh berikut menampilkan konfigurasi analitik untuk bucket dan ID yang ditentukan.

```
aws s3api get-bucket-analytics-configuration \
    --bucket my-bucket \
    --id 1
```

Output:

```
{
    "AnalyticsConfiguration": {
        "StorageClassAnalysis": {},
        "Id": "1"
    }
}
```

- Untuk detail API, lihat GetBucketAnalyticsConfigurationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan detail filter analitik dengan nama 'testfilter' di bucket S3 yang diberikan.

```
Get-S3BucketAnalyticsConfiguration -BucketName 's3testbucket' -AnalyticsId
 'testfilter'
```

- Untuk detail API, lihat GetBucketAnalyticsConfigurationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketCors`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketCors`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        /// <summary>
        /// Retrieve the CORS configuration applied to the Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used
        /// to retrieve the CORS configuration.</param>
        /// <returns>The created CORS configuration object.</returns>
        private static async Task<CORSConfiguration>
 RetrieveCORSConfigurationAsync(AmazonS3Client client)
```

```
        {
            GetCORSConfigurationRequest request = new
 GetCORSConfigurationRequest()
            {
                BucketName = BucketName,
            };
            var response = await client.GetCORSConfigurationAsync(request);
            var configuration = response.Configuration;
            PrintCORSRules(configuration);
            return configuration;
        }
```

- Untuk detail API, lihat GetBucketCorsdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Perintah berikut mengambil konfigurasi Cross-Origin Resource Sharing untuk bucket bernama: `my-bucket`

```
aws s3api get-bucket-cors --bucket my-bucket
```

Output:

```
{
    "CORSRules": [
        {
            "AllowedHeaders": [
                "*"
            ],
            "ExposeHeaders": [
                "x-amz-server-side-encryption"
            ],
            "AllowedMethods": [
                "PUT",
                "POST",
                "DELETE"
            ],
            "MaxAgeSeconds": 3000,
```

```
            "AllowedOrigins": [
                "http://www.example.com"
            ]
        },
        {
            "AllowedHeaders": [
                "Authorization"
            ],
            "MaxAgeSeconds": 3000,
            "AllowedMethods": [
                "GET"
            ],
            "AllowedOrigins": [
                "*"
            ]
        }
    ]
}
```

- Untuk detail API, lihat GetBucketCorsdi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Dapatkan kebijakan CORS untuk bucket.

```
import { GetBucketCorsCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new GetBucketCorsCommand({
    Bucket: "test-bucket",
  });
```

```
  try {
    const { CORSRules } = await client.send(command);
    CORSRules.forEach((cr, i) => {
      console.log(
        `\nCORSRule ${i + 1}`,
        `\n${"-".repeat(10)}`,
        `\nAllowedHeaders: ${cr.AllowedHeaders.join(" ")}`,
        `\nAllowedMethods: ${cr.AllowedMethods.join(" ")}`,
        `\nAllowedOrigins: ${cr.AllowedOrigins.join(" ")}`,
        `\nExposeHeaders: ${cr.ExposeHeaders.join(" ")}`,
        `\nMaxAgeSeconds: ${cr.MaxAgeSeconds}`,
      );
    });
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat GetBucketCorsdi Referensi AWS SDK for JavaScript API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
```

```
        self.bucket = bucket
        self.name = bucket.name


    def get_cors(self):
        """
        Get the CORS rules for the bucket.

        :return The CORS rules for the specified bucket.
        """
        try:
            cors = self.bucket.Cors()
            logger.info(
                "Got CORS rules %s for bucket '%s'.", cors.cors_rules,
self.bucket.name
            )
        except ClientError:
            logger.exception(("Couldn't get CORS for bucket %s.",
self.bucket.name))
            raise
        else:
            return cors
```

- Untuk detail API, lihat GetBucketCorsdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket CORS configuration.
class BucketCorsWrapper
  attr_reader :bucket_cors
```

```
  # @param bucket_cors [Aws::S3::BucketCors] A bucket CORS object configured with
 an existing bucket.
  def initialize(bucket_cors)
    @bucket_cors = bucket_cors
  end

  # Gets the CORS configuration of a bucket.
  #
  # @return [Aws::S3::Type::GetBucketCorsOutput, nil] The current CORS
 configuration for the bucket.
  def get_cors
    @bucket_cors.data
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't get CORS configuration for #{@bucket_cors.bucket.name}. Here's
 why: #{e.message}"
    nil
  end

end
```

- Untuk detail API, lihat GetBucketCorsdi Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketEncryption`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketEncryption`.

CLI

AWS CLI

Untuk mengambil konfigurasi enkripsi sisi server untuk bucket

`get-bucket-encryption`Contoh berikut mengambil konfigurasi enkripsi sisi server untuk bucket. `my-bucket`

```
aws s3api get-bucket-encryption \
```

```
    --bucket my-bucket
```

Output:

```
{
    "ServerSideEncryptionConfiguration": {
        "Rules": [
            {
                "ApplyServerSideEncryptionByDefault": {
                    "SSEAlgorithm": "AES256"
                }
            }
        ]
    }
}
```

- Untuk detail API, lihat GetBucketEncryptiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan semua aturan enkripsi sisi server yang terkait dengan bucket yang diberikan.

```
Get-S3BucketEncryption -BucketName 's3casetestbucket'
```

- Untuk detail API, lihat GetBucketEncryptiondi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketInventoryConfiguration`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketInventoryConfiguration`.

CLI

AWS CLI

Untuk mengambil konfigurasi inventaris untuk bucket

`get-bucket-inventory-configuration`Contoh berikut mengambil konfigurasi inventaris untuk bucket yang ditentukan dengan ID1.

```
aws s3api get-bucket-inventory-configuration \
    --bucket my-bucket \
    --id 1
```

Output:

```
{
    "InventoryConfiguration": {
        "IsEnabled": true,
        "Destination": {
            "S3BucketDestination": {
                "Format": "ORC",
                "Bucket": "arn:aws:s3:::my-bucket",
                "AccountId": "123456789012"
            }
        },
        "IncludedObjectVersions": "Current",
        "Id": "1",
        "Schedule": {
            "Frequency": "Weekly"
        }
    }
}
```

- Untuk detail API, lihat GetBucketInventoryConfigurationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan detail inventaris bernama 'testinventory' untuk bucket S3 yang diberikan.

```
Get-S3BucketInventoryConfiguration -BucketName 's3testbucket' -InventoryId
 'testinventory'
```

- Untuk detail API, lihat GetBucketInventoryConfigurationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketLifecycleConfiguration`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketLifecycleConfiguration`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        /// <summary>
        /// Returns a configuration object for the supplied bucket name.
        /// </summary>
        /// <param name="client">The S3 client object used to call
        /// the GetLifecycleConfigurationAsync method.</param>
        /// <param name="bucketName">The name of the S3 bucket for which a
        /// configuration will be created.</param>
        /// <returns>Returns a new LifecycleConfiguration object.</returns>
        public static async Task<LifecycleConfiguration>
 RetrieveLifecycleConfigAsync(IAmazonS3 client, string bucketName)
        {
            var request = new GetLifecycleConfigurationRequest()
            {
```

```
                BucketName = bucketName,
            };
            var response = await client.GetLifecycleConfigurationAsync(request);
            var configuration = response.Configuration;
            return configuration;
        }
```

- Untuk detail API, lihat GetBucketLifecycleConfigurationdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Perintah berikut mengambil konfigurasi siklus hidup untuk bucket bernama: `my-bucket`

```
aws s3api get-bucket-lifecycle-configuration --bucket my-bucket
```

Output:

```
{
    "Rules": [
        {
            "ID": "Move rotated logs to Glacier",
            "Prefix": "rotated/",
            "Status": "Enabled",
            "Transitions": [
                {
                    "Date": "2015-11-10T00:00:00.000Z",
                    "StorageClass": "GLACIER"
                }
            ]
        },
        {
            "Status": "Enabled",
            "Prefix": "",
            "NoncurrentVersionTransitions": [
                {
                    "NoncurrentDays": 0,
                    "StorageClass": "GLACIER"
                }
```

```
            ],
            "ID": "Move old versions to Glacier"
        }
    ]
}
```

- Untuk detail API, lihat GetBucketLifecycleConfigurationdi Referensi AWS CLI Perintah.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def get_lifecycle_configuration(self):
        """
        Get the lifecycle configuration of the bucket.

        :return: The lifecycle rules of the specified bucket.
        """
        try:
            config = self.bucket.LifecycleConfiguration()
            logger.info(
                "Got lifecycle rules %s for bucket '%s'.",
                config.rules,
```

```
                self.bucket.name,
            )
        except:
            logger.exception(
                "Couldn't get lifecycle rules for bucket '%s'.", self.bucket.name
            )
            raise
        else:
            return config.rules
```

- Untuk detail API, lihat GetBucketLifecycleConfigurationdi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketLocation`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketLocation`.

CLI

AWS CLI

Perintah berikut mengambil batasan lokasi untuk bucket bernama`my-bucket`, jika ada kendala:

```
aws s3api get-bucket-location --bucket my-bucket
```

Output:

```
{
    "LocationConstraint": "us-west-2"
}
```

- Untuk detail API, lihat GetBucketLocationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan batasan lokasi untuk bucket 's3testbucket', jika ada kendala.

```
Get-S3BucketLocation -BucketName 's3testbucket'
```

Output:

```
Value
-----
ap-south-1
```

- Untuk detail API, lihat GetBucketLocationdi Referensi AWS Tools for PowerShell Cmdlet.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
async fn show_buckets(strict: bool, client: &Client, region: &str) -> Result<(),
 Error> {
    let resp = client.list_buckets().send().await?;
    let buckets = resp.buckets();
    let num_buckets = buckets.len();

    let mut in_region = 0;

    for bucket in buckets {
        if strict {
            let r = client
                .get_bucket_location()
                .bucket(bucket.name().unwrap_or_default())
```

```
                .send()
                .await?;

            if r.location_constraint().unwrap().as_ref() == region {
                println!("{}", bucket.name().unwrap_or_default());
                in_region += 1;
            }
        } else {
            println!("{}", bucket.name().unwrap_or_default());
        }
    }

    println!();
    if strict {
        println!(
            "Found {} buckets in the {} region out of a total of {} buckets.",
            in_region, region, num_buckets
        );
    } else {
        println!("Found {} buckets in all regions.", num_buckets);
    }

    Ok(())
}
```

- Untuk detail API, lihat GetBucketLocationreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketLogging`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketLogging`.

CLI

AWS CLI

Untuk mengambil status logging untuk bucket

`get-bucket-logging`Contoh berikut mengambil status logging untuk bucket yang ditentukan.

```
aws s3api get-bucket-logging \
    --bucket my-bucket
```

Output:

```
{
    "LoggingEnabled": {
        "TargetPrefix": "",
        "TargetBucket": "my-bucket-logs"
          }
}
```

- Untuk detail API, lihat GetBucketLoggingdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan status logging untuk bucket yang ditentukan.

```
Get-S3BucketLogging -BucketName 's3testbucket'
```

Output:

```
TargetBucketName   Grants TargetPrefix
----------------   ------ ------------
testbucket1        {}     testprefix
```

- Untuk detail API, lihat GetBucketLoggingdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketMetricsConfiguration`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketMetricsConfiguration`.

CLI

AWS CLI

Untuk mengambil konfigurasi metrik untuk bucket dengan ID tertentu

`get-bucket-metrics-configuration`Contoh berikut menampilkan konfigurasi metrik untuk bucket dan ID yang ditentukan.

```
aws s3api get-bucket-metrics-configuration \
    --bucket my-bucket \
    --id 123
```

Output:

```
{
    "MetricsConfiguration": {
        "Filter": {
            "Prefix": "logs"
        },
        "Id": "123"
    }
}
```

- Untuk detail API, lihat GetBucketMetricsConfigurationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan detail tentang filter metrik bernama 'testfilter' untuk bucket S3 yang diberikan.

```
Get-S3BucketMetricsConfiguration -BucketName 's3testbucket' -MetricsId
 'testfilter'
```

- Untuk detail API, lihat GetBucketMetricsConfigurationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketNotification`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketNotification`.

CLI

AWS CLI

Perintah berikut mengambil konfigurasi notifikasi untuk bucket bernama`my-bucket`:

```
aws s3api get-bucket-notification --bucket my-bucket
```

Output:

```
{
    "TopicConfiguration": {
        "Topic": "arn:aws:sns:us-west-2:123456789012:my-notification-topic",
        "Id": "YmQzMmEwM2EjZWVlI0NGItNzVtZjI1MC00ZjgyLWZDBiZWNl",
        "Event": "s3:ObjectCreated:*",
        "Events": [
            "s3:ObjectCreated:*"
        ]
    }
}
```

- Untuk detail API, lihat GetBucketNotificationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengambil konfigurasi notifikasi dari bucket yang diberikan

```
Get-S3BucketNotification -BucketName kt-tools | select -ExpandProperty
 TopicConfigurations
```

Output:

```
Id   Topic
--   -----
mimo arn:aws:sns:eu-west-1:123456789012:topic-1
```

- Untuk detail API, lihat GetBucketNotificationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketPolicy`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketPolicy`.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::getBucketPolicy(const Aws::String &bucketName,
                                 const Aws::S3::S3ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::GetBucketPolicyRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::GetBucketPolicyOutcome outcome =
            s3Client.GetBucketPolicy(request);
```

```
    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: getBucketPolicy: "
                  << err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    } else {
        Aws::StringStream policy_stream;
        Aws::String line;

        outcome.GetResult().GetPolicy() >> line;
        policy_stream << line;

        std::cout << "Retrieve the policy for bucket '" << bucketName << "':\n\n"
 <<
                  policy_stream.str() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat GetBucketPolicydi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Perintah berikut mengambil kebijakan bucket untuk bucket bernamamy-bucket:

```
aws s3api get-bucket-policy --bucket my-bucket
```

Output:

```
{
    "Policy": "{\"Version\":\"2008-10-17\",\"Statement\":[{\"Sid\":\"\",\"Effect
\":\"Allow\",\"Principal\":\"*\",\"Action\":\"s3:GetObject\",\"Resource\":
\"arn:aws:s3:::my-bucket/*\"},{\"Sid\":\"\",\"Effect\":\"Deny\",\"Principal\":
\"*\",\"Action\":\"s3:GetObject\",\"Resource\":\"arn:aws:s3:::my-bucket/secret/*
\"}]}"
}
```

Dapatkan dan letakkan kebijakan bucket Contoh berikut menunjukkan bagaimana Anda dapat mengunduh kebijakan bucket Amazon S3, membuat modifikasi pada file, dan kemudian `put-bucket-policy` menggunakannya untuk menerapkan kebijakan bucket yang dimodifikasi. Untuk mengunduh kebijakan bucket ke file, Anda dapat menjalankan:

```
aws s3api get-bucket-policy --bucket mybucket --query Policy --output text >
 policy.json
```

Anda kemudian dapat memodifikasi `policy.json` file sesuai kebutuhan. Terakhir, Anda dapat menerapkan kebijakan yang dimodifikasi ini kembali ke bucket S3 dengan menjalankan:

`policy.json`berkas sesuai kebutuhan. Terakhir, Anda dapat menerapkan kebijakan yang dimodifikasi ini kembali ke bucket S3 dengan menjalankan:

berkas sesuai kebutuhan. Terakhir, Anda dapat menerapkan kebijakan yang dimodifikasi ini kembali ke bucket S3 dengan menjalankan:

```
aws s3api put-bucket-policy --bucket mybucket --policy file://policy.json
```

- Untuk detail API, lihat GetBucketPolicydi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetBucketPolicyRequest;
import software.amazon.awssdk.services.s3.model.GetBucketPolicyResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
```

```
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class GetBucketPolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName>

                Where:
                    bucketName - The Amazon S3 bucket to get the policy from.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        System.out.format("Getting policy for bucket: \"%s\"\n\n", bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        String polText = getPolicy(s3, bucketName);
        System.out.println("Policy Text: " + polText);
        s3.close();
    }

    public static String getPolicy(S3Client s3, String bucketName) {
        String policyText;
        System.out.format("Getting policy for bucket: \"%s\"\n\n", bucketName);
        GetBucketPolicyRequest policyReq = GetBucketPolicyRequest.builder()
                .bucket(bucketName)
                .build();

        try {
            GetBucketPolicyResponse policyRes = s3.getBucketPolicy(policyReq);
```

```
            policyText = policyRes.policy();
            return policyText;

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

        return "";
    }
}
```

- Untuk detail API, lihat GetBucketPolicydi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Dapatkan kebijakan bucket.

```
import { GetBucketPolicyCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new GetBucketPolicyCommand({
    Bucket: "test-bucket",
  });

  try {
    const { Policy } = await client.send(command);
    console.log(JSON.parse(Policy));
  } catch (err) {
    console.error(err);
  }
```

```
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat GetBucketPolicydi Referensi AWS SDK for JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun getPolicy(bucketName: String): String? {
    println("Getting policy for bucket $bucketName")

    val request =
        GetBucketPolicyRequest {
            bucket = bucketName
        }

    S3Client { region = "us-east-1" }.use { s3 ->
        val policyRes = s3.getBucketPolicy(request)
        return policyRes.policy
    }
}
```

- Untuk detail API, lihat GetBucketPolicydi AWS SDK untuk referensi API Kotlin.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menampilkan kebijakan bucket yang terkait dengan bucket S3 yang diberikan.

```
Get-S3BucketPolicy -BucketName 's3testbucket'
```

- Untuk detail API, lihat GetBucketPolicydi Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def get_policy(self):
        """
        Get the security policy of the bucket.

        :return: The security policy of the specified bucket, in JSON format.
        """
        try:
            policy = self.bucket.Policy()
            logger.info(
                "Got policy %s for bucket '%s'.", policy.policy, self.bucket.name
            )
        except ClientError:
            logger.exception("Couldn't get policy for bucket '%s'.",
 self.bucket.name)
```

```
            raise
        else:
            return json.loads(policy.policy)
```

- Untuk detail API, lihat GetBucketPolicydi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# Wraps an Amazon S3 bucket policy.
class BucketPolicyWrapper
  attr_reader :bucket_policy

  # @param bucket_policy [Aws::S3::BucketPolicy] A bucket policy object
 configured with an existing bucket.
  def initialize(bucket_policy)
    @bucket_policy = bucket_policy
  end

  # Gets the policy of a bucket.
  #
  # @return [Aws::S3::GetBucketPolicyOutput, nil] The current bucket policy.
  def get_policy
    policy = @bucket_policy.data.policy
    policy.respond_to?(:read) ? policy.read : policy
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't get the policy for #{@bucket_policy.bucket.name}. Here's why:
 #{e.message}"
    nil
  end

end
```

- Untuk detail API, lihat GetBucketPolicydi Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketPolicyStatus`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketPolicyStatus`.

CLI

AWS CLI

Untuk mengambil status kebijakan untuk bucket yang menunjukkan apakah bucket bersifat publik

`get-bucket-policy-status`Contoh berikut mengambil status kebijakan untuk bucket`my-bucket`.

```
aws s3api get-bucket-policy-status \
    --bucket my-bucket
```

Output:

```
{
    "PolicyStatus": {
        "IsPublic": false
    }
}
```

- Untuk detail API, lihat GetBucketPolicyStatusdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan status kebijakan untuk bucket S3 yang diberikan, yang menunjukkan apakah bucket bersifat publik.

```
Get-S3BucketPolicyStatus -BucketName 's3casetestbucket'
```

- Untuk detail API, lihat GetBucketPolicyStatusdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketReplication`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketReplication`.

CLI

AWS CLI

Perintah berikut mengambil konfigurasi replikasi untuk bucket bernama: `my-bucket`

```
aws s3api get-bucket-replication --bucket my-bucket
```

Output:

```
{
    "ReplicationConfiguration": {
        "Rules": [
            {
                "Status": "Enabled",
                "Prefix": "",
                "Destination": {
                    "Bucket": "arn:aws:s3:::my-bucket-backup",
                    "StorageClass": "STANDARD"
                },
                "ID": "ZmUwNzE4ZmQ4tMjVhOS00MTlkLOGI4NDkzZTIWJjNTUtYTA1"
```

```
            }
        ],
        "Role": "arn:aws:iam::123456789012:role/s3-replication-role"
    }
}
```

- Untuk detail API, lihat GetBucketReplicationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Mengembalikan informasi konfigurasi replikasi yang disetel pada bucket bernama 'mybucket'.

```
Get-S3BucketReplication -BucketName mybucket
```

- Untuk detail API, lihat GetBucketReplicationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketRequestPayment`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketRequestPayment`.

CLI

AWS CLI

Untuk mengambil konfigurasi pembayaran permintaan untuk bucket

`get-bucket-request-payment`Contoh berikut mengambil konfigurasi requester pay untuk bucket yang ditentukan.

```
aws s3api get-bucket-request-payment \
    --bucket my-bucket
```

Output:

```
{
    "Payer": "BucketOwner"
}
```

- Untuk detail API, lihat GetBucketRequestPaymentdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Mengembalikan konfigurasi permintaan pembayaran untuk bucket bernama 'mybucket'. Secara default, pemilik bucket membayar unduhan dari bucket.

```
Get-S3BucketRequestPayment -BucketName mybucket
```

- Untuk detail API, lihat GetBucketRequestPaymentdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketTagging`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketTagging`.

CLI

AWS CLI

Perintah berikut mengambil konfigurasi penandaan untuk bucket bernama: `my-bucket`

```
aws s3api get-bucket-tagging --bucket my-bucket
```

Output:

```
{
    "TagSet": [
        {
            "Value": "marketing",
```

```
            "Key": "organization"
        }
    ]
}
```

- Untuk detail API, lihat GetBucketTaggingdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan semua tag yang terkait dengan bucket yang diberikan.

```
Get-S3BucketTagging -BucketName 's3casetestbucket'
```

- Untuk detail API, lihat GetBucketTaggingdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketVersioning`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketVersioning`.

CLI

AWS CLI

Perintah berikut mengambil konfigurasi pembuatan versi untuk bucket bernama: `my-bucket`

```
aws s3api get-bucket-versioning --bucket my-bucket
```

Output:

```
{
    "Status": "Enabled"
}
```

- Untuk detail API, lihat GetBucketVersioningdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan status pembuatan versi sehubungan dengan bucket yang diberikan.

```
Get-S3BucketVersioning -BucketName 's3testbucket'
```

- Untuk detail API, lihat GetBucketVersioningdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetBucketWebsite`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetBucketWebsite`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
                // Get the website configuration.
                GetBucketWebsiteRequest getRequest = new
GetBucketWebsiteRequest()
                {
                    BucketName = bucketName,
                };
                GetBucketWebsiteResponse getResponse = await
client.GetBucketWebsiteAsync(getRequest);
                Console.WriteLine($"Index document:
{getResponse.WebsiteConfiguration.IndexDocumentSuffix}");
                Console.WriteLine($"Error document:
{getResponse.WebsiteConfiguration.ErrorDocument}");
```

- Untuk detail API, lihat GetBucketWebsitedi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::getWebsiteConfig(const Aws::String &bucketName,
                                  const Aws::S3::S3ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::GetBucketWebsiteRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::GetBucketWebsiteOutcome outcome =
            s3Client.GetBucketWebsite(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();

        std::cerr << "Error: GetBucketWebsite: "
                  << err.GetMessage() << std::endl;
    } else {
        Aws::S3::Model::GetBucketWebsiteResult websiteResult =
 outcome.GetResult();

        std::cout << "Success: GetBucketWebsite: "
                  << std::endl << std::endl
                  << "For bucket '" << bucketName << "':"
                  << std::endl
                  << "Index page : "
                  << websiteResult.GetIndexDocument().GetSuffix()
                  << std::endl
```

```
                    << "Error page: "
                    << websiteResult.GetErrorDocument().GetKey()
                    << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat GetBucketWebsitedi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Perintah berikut mengambil konfigurasi situs web statis untuk bucket bernamamy-bucket:

```
aws s3api get-bucket-website --bucket my-bucket
```

Output:

```
{
    "IndexDocument": {
        "Suffix": "index.html"
    },
    "ErrorDocument": {
        "Key": "error.html"
    }
}
```

- Untuk detail API, lihat GetBucketWebsitedi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Dapatkan konfigurasi situs web.

```
import { GetBucketWebsiteCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new GetBucketWebsiteCommand({
    Bucket: "test-bucket",
  });

  try {
    const { ErrorDocument, IndexDocument } = await client.send(command);
    console.log(
      `Your bucket is set up to host a website. It has an error document:`,
      `${ErrorDocument.Key}, and an index document: ${IndexDocument.Suffix}.`,
    );
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk detail API, lihat GetBucketWebsitedi Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan detail konfigurasi situs web statis dari bucket S3 yang diberikan.

```
Get-S3BucketWebsite -BucketName 's3testbucket'
```

- Untuk detail API, lihat GetBucketWebsitedi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetObject`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetObject`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Mendapatkan objek dari bucket jika telah diubah
- Dapatkan objek dari Titik Akses Multi-Region
- Memulai bucket dan objek
- Memulai enkripsi
- Lacak unggahan dan unduhan

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
/// <summary>
/// Shows how to download an object from an Amazon S3 bucket to the
/// local computer.
/// </summary>
/// <param name="client">An initialized Amazon S3 client object.</param>
/// <param name="bucketName">The name of the bucket where the object is
/// currently stored.</param>
/// <param name="objectName">The name of the object to download.</param>
/// <param name="filePath">The path, including filename, where the
/// downloaded object will be stored.</param>
/// <returns>A boolean value indicating the success or failure of the
/// download process.</returns>
public static async Task<bool> DownloadObjectFromBucketAsync(
    IAmazonS3 client,
    string bucketName,
    string objectName,
```

```
            string filePath)
        {
            // Create a GetObject request
            var request = new GetObjectRequest
            {
                BucketName = bucketName,
                Key = objectName,
            };

            // Issue request and remember to dispose of the response
            using GetObjectResponse response = await
 client.GetObjectAsync(request);

            try
            {
                // Save object to local file
                await response.WriteResponseStreamToFileAsync($"{filePath}\
\{objectName}", true, CancellationToken.None);
                return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error saving {objectName}: {ex.Message}");
                return false;
            }
        }
```

- Untuk detail API, lihat GetObjectdi Referensi AWS SDK for .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
###############################################################################
```

```
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function download_object_from_bucket
#
# This function downloads an object in a bucket to a file.
#
# Parameters:
#       $1 - The name of the bucket to download the object from.
#       $2 - The path and file name to store the downloaded bucket.
#       $3 - The key (name) of the object in the bucket.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function download_object_from_bucket() {
  local bucket_name=$1
  local destination_file_name=$2
  local object_name=$3
  local response

  response=$(aws s3api get-object \
    --bucket "$bucket_name" \
    --key "$object_name" \
    "$destination_file_name")

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports put-object operation failed.\n$response"
    return 1
  fi
}
```

- Untuk detail API, lihat GetObjectdi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::getObject(const Aws::String &objectKey,
                           const Aws::String &fromBucket,
                           const Aws::S3::S3ClientConfiguration &clientConfig) {
    Aws::S3::S3Client client(clientConfig);

    Aws::S3::Model::GetObjectRequest request;
    request.SetBucket(fromBucket);
    request.SetKey(objectKey);

    Aws::S3::Model::GetObjectOutcome outcome =
            client.GetObject(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: getObject: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    } else {
        std::cout << "Successfully retrieved '" << objectKey << "' from '"
                  << fromBucket << "'." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat GetObjectdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Contoh berikut menggunakan `get-object` perintah untuk mengunduh objek dari Amazon S3:

```
aws s3api get-object --bucket text-content --key dir/my_images.tar.bz2
 my_images.tar.bz2
```

Perhatikan bahwa parameter outfile ditentukan tanpa nama opsi seperti “--outfile”. Nama file output harus menjadi parameter terakhir dalam perintah.

Contoh di bawah ini menunjukkan penggunaan `--range` untuk men-download rentang byte tertentu dari sebuah objek. Perhatikan rentang byte perlu diawali dengan “byte =”:

```
aws s3api get-object --bucket text-content --key dir/my_data --range
 bytes=8888-9999 my_data_range
```

Untuk informasi selengkapnya tentang mengambil objek, lihat Mendapatkan Objek di Panduan Pengembang Amazon S3.

- Untuk detail API, lihat GetObjectdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
```

```
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// DownloadFile gets an object from a bucket and stores it in a local file.
func (basics BucketBasics) DownloadFile(bucketName string, objectKey string,
 fileName string) error {
 result, err := basics.S3Client.GetObject(context.TODO(), &s3.GetObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't get object %v:%v. Here's why: %v\n", bucketName,
 objectKey, err)
  return err
 }
 defer result.Body.Close()
 file, err := os.Create(fileName)
 if err != nil {
  log.Printf("Couldn't create file %v. Here's why: %v\n", fileName, err)
  return err
 }
 defer file.Close()
 body, err := io.ReadAll(result.Body)
 if err != nil {
  log.Printf("Couldn't read object body from %v. Here's why: %v\n", objectKey,
 err)
 }
 _, err = file.Write(body)
 return err
}
```

- Untuk detail API, lihat GetObjectdi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Membaca data sebagai array byte menggunakan S3Client.

```
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class GetObjectData {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName> <path>

                Where:
                    bucketName - The Amazon S3 bucket name.\s
                    keyName - The key name.\s
```

```
                    path - The path where the file is written to.\s
                """";

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        String path = args[2];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        getObjectBytes(s3, bucketName, keyName, path);
    }

    public static void getObjectBytes(S3Client s3, String bucketName, String
keyName, String path) {
        try {
            GetObjectRequest objectRequest = GetObjectRequest
                    .builder()
                    .key(keyName)
                    .bucket(bucketName)
                    .build();

            ResponseBytes<GetObjectResponse> objectBytes =
s3.getObjectAsBytes(objectRequest);
            byte[] data = objectBytes.asByteArray();

            // Write the data to a local file.
            File myFile = new File(path);
            OutputStream os = new FileOutputStream(myFile);
            os.write(data);
            System.out.println("Successfully obtained bytes from an S3 object");
            os.close();

        } catch (IOException ex) {
            ex.printStackTrace();
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
```

```
        }
    }
}
```

Gunakan S3 TransferManager untuk mengunduh objek dalam bucket S3 ke file lokal. Lihat file lengkap dan lakukan pengujian.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedFileDownload;
import software.amazon.awssdk.transfer.s3.model.DownloadFileRequest;
import software.amazon.awssdk.transfer.s3.model.FileDownload;
import software.amazon.awssdk.transfer.s3.progress.LoggingTransferListener;

import java.io.IOException;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.UUID;

    public Long downloadFile(S3TransferManager transferManager, String
 bucketName,
                             String key, String downloadedFileWithPath) {
        DownloadFileRequest downloadFileRequest = DownloadFileRequest.builder()
                .getObjectRequest(b -> b.bucket(bucketName).key(key))
                .destination(Paths.get(downloadedFileWithPath))
                .build();

        FileDownload downloadFile =
 transferManager.downloadFile(downloadFileRequest);

        CompletedFileDownload downloadResult =
 downloadFile.completionFuture().join();
        logger.info("Content length [{}]",
 downloadResult.response().contentLength());
        return downloadResult.response().contentLength();
    }
```

Baca tanda milik objek menggunakan S3Client.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectTaggingRequest;
import software.amazon.awssdk.services.s3.model.GetObjectTaggingResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.Tag;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class GetObjectTags {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName>\s

                Where:
                    bucketName - The Amazon S3 bucket name.\s
                    keyName - A key name that represents the object.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
```

```
                .region(region)
                .build();

        listTags(s3, bucketName, keyName);
        s3.close();
    }

    public static void listTags(S3Client s3, String bucketName, String keyName) {
        try {
            GetObjectTaggingRequest getTaggingRequest = GetObjectTaggingRequest
                    .builder()
                    .key(keyName)
                    .bucket(bucketName)
                    .build();

            GetObjectTaggingResponse tags =
 s3.getObjectTagging(getTaggingRequest);
            List<Tag> tagSet = tags.tagSet();
            for (Tag tag : tagSet) {
                System.out.println(tag.key());
                System.out.println(tag.value());
            }

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

Dapatkan URL untuk objek menggunakan S3Client.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetUrlRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.net.URL;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
```

```
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class GetObjectUrl {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName>\s

                Where:
                    bucketName - The Amazon S3 bucket name.
                    keyName - A key name that represents the object.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        getURL(s3, bucketName, keyName);
        s3.close();
    }

    public static void getURL(S3Client s3, String bucketName, String keyName) {
        try {
            GetUrlRequest request = GetUrlRequest.builder()
                    .bucket(bucketName)
                    .key(keyName)
                    .build();

            URL url = s3.utilities().getUrl(request);
            System.out.println("The URL for  " + keyName + " is " + url);
```

```
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

Dapatkan objek dengan menggunakan objek klien S3Presigner menggunakan S3Client.

```
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.time.Duration;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import
 software.amazon.awssdk.services.s3.presigner.model.GetObjectPresignRequest;
import
 software.amazon.awssdk.services.s3.presigner.model.PresignedGetObjectRequest;
import software.amazon.awssdk.services.s3.presigner.S3Presigner;
import software.amazon.awssdk.utils.IoUtils;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class GetObjectPresignedUrl {
    public static void main(String[] args) {
        final String USAGE = """

                Usage:
                    <bucketName> <keyName>\s

                Where:
                    bucketName - The Amazon S3 bucket name.\s
```

```
                    keyName - A key name that represents a text file.\s
                """";

        if (args.length != 2) {
            System.out.println(USAGE);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        Region region = Region.US_EAST_1;
        S3Presigner presigner = S3Presigner.builder()
                .region(region)
                .build();

        getPresignedUrl(presigner, bucketName, keyName);
        presigner.close();
    }

    public static void getPresignedUrl(S3Presigner presigner, String bucketName,
String keyName) {
        try {
            GetObjectRequest getObjectRequest = GetObjectRequest.builder()
                    .bucket(bucketName)
                    .key(keyName)
                    .build();

            GetObjectPresignRequest getObjectPresignRequest =
GetObjectPresignRequest.builder()
                    .signatureDuration(Duration.ofMinutes(60))
                    .getObjectRequest(getObjectRequest)
                    .build();

            PresignedGetObjectRequest presignedGetObjectRequest =
presigner.presignGetObject(getObjectPresignRequest);
            String theUrl = presignedGetObjectRequest.url().toString();
            System.out.println("Presigned URL: " + theUrl);
            HttpURLConnection connection = (HttpURLConnection)
presignedGetObjectRequest.url().openConnection();
            presignedGetObjectRequest.httpRequest().headers().forEach((header,
values) -> {
                values.forEach(value -> {
                    connection.addRequestProperty(header, value);
                });
```

```
            });

            // Send any request payload that the service needs (not needed when
            // isBrowserExecutable is true).
            if (presignedGetObjectRequest.signedPayload().isPresent()) {
                connection.setDoOutput(true);

                try (InputStream signedPayload =
 presignedGetObjectRequest.signedPayload().get().asInputStream();
                        OutputStream httpOutputStream =
 connection.getOutputStream()) {
                    IoUtils.copy(signedPayload, httpOutputStream);
                }
            }

            // Download the result of executing the request.
            try (InputStream content = connection.getInputStream()) {
                System.out.println("Service returned response: ");
                IoUtils.copy(content, System.out);
            }

        } catch (S3Exception | IOException e) {
            e.getStackTrace();
        }
    }
}
```

Dapatkan objek dengan menggunakan ResponseTransformer objek dan S3Client.

```
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.core.sync.ResponseTransformer;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;

/**
```

```
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class GetDataResponseTransformer {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName> <path>

                Where:
                    bucketName - The Amazon S3 bucket name.\s
                    keyName - The key name.\s
                    path - The path where the file is written to.\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        String path = args[2];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        getObjectBytes(s3, bucketName, keyName, path);
        s3.close();
    }

    public static void getObjectBytes(S3Client s3, String bucketName, String
 keyName, String path) {
        try {
            GetObjectRequest objectRequest = GetObjectRequest
                    .builder()
```

```
                    .key(keyName)
                    .bucket(bucketName)
                    .build();

            ResponseBytes<GetObjectResponse> objectBytes =
 s3.getObject(objectRequest, ResponseTransformer.toBytes());
            byte[] data = objectBytes.asByteArray();

            // Write the data to a local file.
            File myFile = new File(path);
            OutputStream os = new FileOutputStream(myFile);
            os.write(data);
            System.out.println("Successfully obtained bytes from an S3 object");
            os.close();

        } catch (IOException ex) {
            ex.printStackTrace();
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat GetObjectdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Unduh objek tersebut.

```
import { GetObjectCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});
```

```
export const main = async () => {
  const command = new GetObjectCommand({
    Bucket: "test-bucket",
    Key: "hello-s3.txt",
  });

  try {
    const response = await client.send(command);
    // The Body object also has 'transformToByteArray' and 'transformToWebStream'
 methods.
    const str = await response.Body.transformToString();
    console.log(str);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat GetObjectdi Referensi AWS SDK for JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun getObjectBytes(
    bucketName: String,
    keyName: String,
    path: String,
) {
    val request =
        GetObjectRequest {
            key = keyName
            bucket = bucketName
```

```
        }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.getObject(request) { resp ->
            val myFile = File(path)
            resp.body?.writeToFile(myFile)
            println("Successfully read $keyName from $bucketName")
        }
    }
}
```

- Untuk detail API, lihat GetObjectdi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Dapatkan objek.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        try {
            $file = $this->s3client->getObject([
                'Bucket' => $this->bucketName,
                'Key' => $fileName,
            ]);
            $body = $file->get('Body');
            $body->rewind();
            echo "Downloaded the file and it begins with: {$body->read(26)}.\n";
        } catch (Exception $exception) {
            echo "Failed to download $fileName from $this->bucketName with error:
 " . $exception->getMessage();
            exit("Please fix error with file downloading before continuing.");
        }
```

- Untuk detail API, lihat GetObjectdi Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengambil item "sample.txt" dari bucket “test-files” dan menyimpannya ke file bernama "local-sample.txt" di lokasi saat ini. File "local-sample.txt" tidak harus ada sebelum perintah ini dipanggil.

```
Read-S3Object -BucketName test-files -Key sample.txt -File local-sample.txt
```

Contoh 2: Perintah ini mengambil direktori virtual “DIR” dari bucket “test-files” dan menyimpannya ke folder bernama “Local-dir” di lokasi saat ini. Folder “Local-dir” tidak harus ada sebelum perintah ini dipanggil.

```
Read-S3Object -BucketName test-files -KeyPrefix DIR -Folder Local-DIR
```

Contoh 3: Mengunduh semua objek dengan kunci yang diakhiri dengan '.json' dari ember dengan 'konfigurasi' dalam nama ember ke file di folder yang ditentukan. Kunci objek digunakan untuk mengatur nama file.

```
Get-S3Bucket | ? { $_.BucketName -like '*config*' } | Get-S3Object | ? { $_.Key -
like '*.json' } | Read-S3Object -Folder C:\ConfigObjects
```

- Untuk detail API, lihat GetObjectdi Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    def get(self):
        """
        Gets the object.

        :return: The object data in bytes.
        """
        try:
            body = self.object.get()["Body"].read()
            logger.info(
                "Got object '%s' from bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
        except ClientError:
            logger.exception(
                "Couldn't get object '%s' from bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
            raise
        else:
            return body
```

- Untuk detail API, lihat GetObjectdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Dapatkan objek.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectGetWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Gets the object directly to a file.
  #
  # @param target_path [String] The path to the file where the object is
 downloaded.
  # @return [Aws::S3::Types::GetObjectOutput, nil] The retrieved object data if
 successful; otherwise nil.
  def get_object(target_path)
    @object.get(response_target: target_path)
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't get object #{@object.key}. Here's why: #{e.message}"
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object.txt"
  target_path = "my-object-as-file.txt"

  wrapper = ObjectGetWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
```

```
  obj_data = wrapper.get_object(target_path)
  return unless obj_data

  puts "Object #{object_key} (#{obj_data.content_length} bytes} downloaded to
 #{target_path}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Dapatkan objek dan laporkan status enkripsi di sisi servernya.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectGetEncryptionWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Gets the object into memory.
  #
  # @return [Aws::S3::Types::GetObjectOutput, nil] The retrieved object data if
 successful; otherwise nil.
  def get_object
    @object.get
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't get object #{@object.key}. Here's why: #{e.message}"
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object.txt"

  wrapper = ObjectGetEncryptionWrapper.new(Aws::S3::Object.new(bucket_name,
 object_key))
  obj_data = wrapper.get_object
  return unless obj_data
```

```
  encryption = obj_data.server_side_encryption.nil? ? "no" :
 obj_data.server_side_encryption
  puts "Object #{object_key} uses #{encryption} encryption."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk detail API, lihat GetObjectdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
async fn get_object(client: Client, opt: Opt) -> Result<usize, anyhow::Error> {
    trace!("bucket:      {}", opt.bucket);
    trace!("object:      {}", opt.object);
    trace!("destination: {}", opt.destination.display());

    let mut file = File::create(opt.destination.clone())?;

    let mut object = client
        .get_object()
        .bucket(opt.bucket)
        .key(opt.object)
        .send()
        .await?;

    let mut byte_count = 0_usize;
    while let Some(bytes) = object.body.try_next().await? {
        let bytes_len = bytes.len();
        file.write_all(&bytes)?;
        trace!("Intermediate write of {bytes_len}");
        byte_count += bytes_len;
    }
```

```
    Ok(byte_count)
}
```

- Untuk detail API, lihat GetObjectreferensi AWS SDK for Rust API.

SAP ABAP

SDK untuk SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    TRY.
        oo_result = lo_s3->getobject(           " oo_result is returned for
testing purposes. "
                  iv_bucket = iv_bucket_name
                  iv_key = iv_object_key
               ).
        DATA(lv_object_data) = oo_result->get_body( ).
        MESSAGE 'Object retrieved from S3 bucket.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
      CATCH /aws1/cx_s3_nosuchkey.
        MESSAGE 'Object key does not exist.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat GetObjectdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK untuk Swift

> **Note**
>
> Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Unduh objek dari bucket ke file lokal.

```
    public func downloadFile(bucket: String, key: String, to: String) async
throws {
        let fileUrl = URL(fileURLWithPath: to).appendingPathComponent(key)

        let input = GetObjectInput(
            bucket: bucket,
            key: key
        )
        let output = try await client.getObject(input: input)

        // Get the data stream object. Return immediately if there isn't one.
        guard let body = output.body,
              let data = try await body.readData() else {
            return
        }
        try data.write(to: fileUrl)
    }
```

Baca objek ke objek Swift Data.

```
    public func readFile(bucket: String, key: String) async throws -> Data {
        let input = GetObjectInput(
```

```
            bucket: bucket,
            key: key
        )
        let output = try await client.getObject(input: input)

        // Get the stream and return its contents in a `Data` object. If
        // there is no stream, return an empty `Data` object instead.
        guard let body = output.body,
              let data = try await body.readData() else {
            return "".data(using: .utf8)!
        }

        return data
    }
```

- Untuk detail API, lihat referensi GetObject AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetObjectAcl`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetObjectAcl`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Mengelola daftar kontrol akses (ACL)

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::getObjectAcl(const Aws::String &bucketName,
                              const Aws::String &objectKey,
                              const Aws::S3::S3ClientConfiguration &clientConfig)
 {
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::GetObjectAclRequest request;
    request.SetBucket(bucketName);
    request.SetKey(objectKey);

    Aws::S3::Model::GetObjectAclOutcome outcome =
            s3Client.GetObjectAcl(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: getObjectAcl: "
                  << err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    } else {
        Aws::Vector<Aws::S3::Model::Grant> grants =
                outcome.GetResult().GetGrants();

        for (auto it = grants.begin(); it != grants.end(); it++) {
            std::cout << "For object " << objectKey << ": "
                      << std::endl << std::endl;

            Aws::S3::Model::Grant grant = *it;
            Aws::S3::Model::Grantee grantee = grant.GetGrantee();

            if (grantee.TypeHasBeenSet()) {
                std::cout << "Type:          "
                          << getGranteeTypeString(grantee.GetType()) <<
 std::endl;
            }

            if (grantee.DisplayNameHasBeenSet()) {
                std::cout << "Display name:  "
                          << grantee.GetDisplayName() << std::endl;
            }

            if (grantee.EmailAddressHasBeenSet()) {
                std::cout << "Email address: "
                          << grantee.GetEmailAddress() << std::endl;
```

```
            }

            if (grantee.IDHasBeenSet()) {
                std::cout << "ID:            "
                          << grantee.GetID() << std::endl;
            }

            if (grantee.URIHasBeenSet()) {
                std::cout << "URI:           "
                          << grantee.GetURI() << std::endl;
            }

            std::cout << "Permission:    " <<
                      getPermissionString(grant.GetPermission()) <<
                      std::endl << std::endl;
        }
    }

    return outcome.IsSuccess();
}

//! Routine which converts a built-in type enumeration to a human-readable
 string.
/*!
 \param type: Type enumeration.
 \return String: Human-readable string
*/
Aws::String getGranteeTypeString(const Aws::S3::Model::Type &type) {
    switch (type) {
        case Aws::S3::Model::Type::AmazonCustomerByEmail:
            return "Email address of an AWS account";
        case Aws::S3::Model::Type::CanonicalUser:
            return "Canonical user ID of an AWS account";
        case Aws::S3::Model::Type::Group:
            return "Predefined Amazon S3 group";
        case Aws::S3::Model::Type::NOT_SET:
            return "Not set";
        default:
            return "Type unknown";
    }
}

//! Routine which converts a built-in type enumeration to a human-readable
 string.
```

```
/*!
 \param permission: Permission enumeration.
 \return String: Human-readable string
*/
Aws::String getPermissionString(const Aws::S3::Model::Permission &permission) {
    switch (permission) {
        case Aws::S3::Model::Permission::FULL_CONTROL:
            return "Can read this object's data and its metadata, "
                   "and read/write this object's permissions";
        case Aws::S3::Model::Permission::NOT_SET:
            return "Permission not set";
        case Aws::S3::Model::Permission::READ:
            return "Can read this object's data and its metadata";
        case Aws::S3::Model::Permission::READ_ACP:
            return "Can read this object's permissions";
            // case Aws::S3::Model::Permission::WRITE // Not applicable.
        case Aws::S3::Model::Permission::WRITE_ACP:
            return "Can write this object's permissions";
        default:
            return "Permission unknown";
    }
}
```

- Untuk detail API, lihat GetObjectAcldi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Perintah berikut mengambil daftar kontrol akses untuk objek dalam bucket bernamamy-bucket:

```
aws s3api get-object-acl --bucket my-bucket --key index.html
```

Output:

```
{
    "Owner": {
        "DisplayName": "my-username",
        "ID": "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
```

```
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "my-username",
                "ID":
 "7009a8971cd538e11f6b6606438875e7c86c5b672f46db45460ddcd087d36c32"
            },
            "Permission": "FULL_CONTROL"
        },
        {
            "Grantee": {
                "URI": "http://acs.amazonaws.com/groups/global/AllUsers"
            },
            "Permission": "READ"
        }
    ]
}
```

- Untuk detail API, lihat GetObjectAcldi Referensi AWS CLI Perintah.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun getBucketACL(
    objectKey: String,
    bucketName: String,
) {
    val request =
        GetObjectAclRequest {
            bucket = bucketName
            key = objectKey
        }

    S3Client { region = "us-east-1" }.use { s3 ->
```

```
        val response = s3.getObjectAcl(request)
        response.grants?.forEach { grant ->
            println("Grant permission is ${grant.permission}")
        }
    }
}
```

- Untuk detail API, lihat GetObjectAcldi AWS SDK untuk referensi API Kotlin.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    def get_acl(self):
        """
        Gets the ACL of the object.

        :return: The ACL of the object.
        """
        try:
            acl = self.object.Acl()
            logger.info(
```

```
                "Got ACL for object %s owned by %s.",
                self.object.key,
                acl.owner["DisplayName"],
            )
        except ClientError:
            logger.exception("Couldn't get ACL for object %s.", self.object.key)
            raise
        else:
            return acl
```

- Untuk detail API, lihat GetObjectAcldi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetObjectLegalHold`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetObjectLegalHold`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Dapatkan konfigurasi penahanan hukum suatu objek
- Kunci objek Amazon S3

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
```

```
    /// Get the legal hold details for an S3 object.
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The object key.</param>
    /// <returns>The object legal hold details.</returns>
    public async Task<ObjectLockLegalHold> GetObjectLegalHold(string bucketName,
        string objectKey)
    {
        try
        {
            var request = new GetObjectLegalHoldRequest()
            {
                BucketName = bucketName,
                Key = objectKey
            };

            var response = await _amazonS3.GetObjectLegalHoldAsync(request);
            Console.WriteLine($"\tObject legal hold for {objectKey} in
{bucketName}: " +
                              $"\n\tStatus: {response.LegalHold.Status}");
            return response.LegalHold;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to fetch legal hold: '{ex.Message}'");
            return new ObjectLockLegalHold();
        }
    }
```

- Untuk detail API, lihat GetObjectLegalHolddi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Mengambil status Legal Hold dari suatu objek

`get-object-legal-hold`Contoh berikut mengambil status Penahanan Hukum untuk objek yang ditentukan.

```
aws s3api get-object-legal-hold \
    --bucket my-bucket-with-object-lock \
```

```
    --key doc1.rtf
```

Output:

```
{
    "LegalHold": {
        "Status": "ON"
    }
}
```

- Untuk detail API, lihat GetObjectLegalHolddi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// S3Actions wraps S3 service actions.
type S3Actions struct {
 S3Client  *s3.Client
 S3Manager *manager.Uploader
}



// GetObjectLegalHold retrieves the legal hold status for an S3 object.
func (actor S3Actions) GetObjectLegalHold(ctx context.Context, bucket string, key
 string, versionId string) (*types.ObjectLockLegalHoldStatus, error) {
 var status *types.ObjectLockLegalHoldStatus
 input := &s3.GetObjectLegalHoldInput{
  Bucket:    aws.String(bucket),
  Key:       aws.String(key),
  VersionId: aws.String(versionId),
 }
```

```
 output, err := actor.S3Client.GetObjectLegalHold(ctx, input)
 if err != nil {
  var noSuchKeyErr *types.NoSuchKey
  var apiErr *smithy.GenericAPIError
  if errors.As(err, &noSuchKeyErr) {
   log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
   err = noSuchKeyErr
  } else if errors.As(err, &apiErr) {
   switch apiErr.ErrorCode() {
   case "NoSuchObjectLockConfiguration":
    log.Printf("Object %s does not have an object lock configuration.\n", key)
    err = nil
   case "InvalidRequest":
    log.Printf("Bucket %s does not have an object lock configuration.\n", bucket)
    err = nil
   }
  }
 } else {
  status = &output.LegalHold.Status
 }

 return status, err
}
```

- Untuk detail API, lihat GetObjectLegalHolddi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    // Get the legal hold details for an S3 object.
    public ObjectLockLegalHold getObjectLegalHold(String bucketName, String
 objectKey) {
        try {
```

```
            GetObjectLegalHoldRequest legalHoldRequest =
GetObjectLegalHoldRequest.builder()
                .bucket(bucketName)
                .key(objectKey)
                .build();

            GetObjectLegalHoldResponse response =
getClient().getObjectLegalHold(legalHoldRequest);
            System.out.println("Object legal hold for " + objectKey + " in " +
bucketName +
                ":\n\tStatus: " + response.legalHold().status());
            return response.legalHold();

        } catch (S3Exception ex) {
            System.out.println("\tUnable to fetch legal hold: '" +
ex.getMessage() + "'");
        }

        return null;
    }
```

- Untuk detail API, lihat GetObjectLegalHolddi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetObjectLockConfiguration`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetObjectLockConfiguration`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kunci objek Amazon S3

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
   /// <summary>
   /// Get the object lock configuration details for an S3 bucket.
   /// </summary>
   /// <param name="bucketName">The bucket to get details.</param>
   /// <returns>The bucket's object lock configuration details.</returns>
   public async Task<ObjectLockConfiguration>
GetBucketObjectLockConfiguration(string bucketName)
   {
       try
       {
           var request = new GetObjectLockConfigurationRequest()
           {
               BucketName = bucketName
           };

           var response = await
_amazonS3.GetObjectLockConfigurationAsync(request);
           Console.WriteLine($"\tBucket object lock config for {bucketName} in
{bucketName}: " +
                             $"\n\tEnabled:
{response.ObjectLockConfiguration.ObjectLockEnabled}" +
                             $"\n\tRule:
{response.ObjectLockConfiguration.Rule?.DefaultRetention}");

           return response.ObjectLockConfiguration;
       }
       catch (AmazonS3Exception ex)
       {
           Console.WriteLine($"\tUnable to fetch object lock config:
'{ex.Message}'");
           return new ObjectLockConfiguration();
       }
```

```
    }
```

- Untuk detail API, lihat GetObjectLockConfigurationdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk mengambil konfigurasi kunci objek untuk bucket

`get-object-lock-configuration`Contoh berikut mengambil konfigurasi kunci objek untuk bucket yang ditentukan.

```
aws s3api get-object-lock-configuration \
    --bucket my-bucket-with-object-lock
```

Output:

```
{
    "ObjectLockConfiguration": {
        "ObjectLockEnabled": "Enabled",
        "Rule": {
            "DefaultRetention": {
                "Mode": "COMPLIANCE",
                "Days": 50
            }
        }
    }
}
```

- Untuk detail API, lihat GetObjectLockConfigurationdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// S3Actions wraps S3 service actions.
type S3Actions struct {
 S3Client  *s3.Client
 S3Manager *manager.Uploader
}



// GetObjectLockConfiguration retrieves the object lock configuration for an S3
 bucket.
func (actor S3Actions) GetObjectLockConfiguration(ctx context.Context, bucket
 string) (*types.ObjectLockConfiguration, error) {
 var lockConfig *types.ObjectLockConfiguration
 input := &s3.GetObjectLockConfigurationInput{
  Bucket: aws.String(bucket),
 }

 output, err := actor.S3Client.GetObjectLockConfiguration(ctx, input)
 if err != nil {
  var noBucket *types.NoSuchBucket
  var apiErr *smithy.GenericAPIError
  if errors.As(err, &noBucket) {
   log.Printf("Bucket %s does not exist.\n", bucket)
   err = noBucket
  } else if errors.As(err, &apiErr) && apiErr.ErrorCode() ==
 "ObjectLockConfigurationNotFoundError" {
   log.Printf("Bucket %s does not have an object lock configuration.\n", bucket)
   err = nil
  }
 } else {
  lockConfig = output.ObjectLockConfiguration
```

```
 }

 return lockConfig, err
}
```

- Untuk detail API, lihat GetObjectLockConfigurationdi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
   // Get the object lock configuration details for an S3 bucket.
   public void getBucketObjectLockConfiguration(String bucketName) {
       GetObjectLockConfigurationRequest objectLockConfigurationRequest =
GetObjectLockConfigurationRequest.builder()
           .bucket(bucketName)
           .build();

       GetObjectLockConfigurationResponse response =
getClient().getObjectLockConfiguration(objectLockConfigurationRequest);
       System.out.println("Bucket object lock config for "+bucketName +":  ");
       System.out.println("\tEnabled:
"+response.objectLockConfiguration().objectLockEnabled());
       System.out.println("\tRule: "+
response.objectLockConfiguration().rule().defaultRetention());
   }
```

- Untuk detail API, lihat GetObjectLockConfigurationdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { fileURLToPath } from "url";
import {
  GetObjectLockConfigurationCommand,
  S3Client,
} from "@aws-sdk/client-s3";

/**
 * @param {S3Client} client
 * @param {string} bucketName
 */
export const main = async (client, bucketName) => {
  const command = new GetObjectLockConfigurationCommand({
    Bucket: bucketName,
    // Optionally, you can provide additional parameters
    // ExpectedBucketOwner: "ACCOUNT_ID",
  });

  try {
    const { ObjectLockConfiguration } = await client.send(command);
    console.log(`Object Lock Configuration: ${ObjectLockConfiguration}`);
  } catch (err) {
    console.error(err);
  }
};

// Invoke main function if this file was run directly.
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  main(new S3Client(), "BUCKET_NAME");
}
```

- Untuk detail API, lihat GetObjectLockConfigurationdi Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan nilai 'Diaktifkan' jika konfigurasi kunci Objek diaktifkan untuk bucket S3 yang diberikan.

```
Get-S3ObjectLockConfiguration -BucketName 's3buckettesting' -Select
 ObjectLockConfiguration.ObjectLockEnabled
```

Output:

```
Value
-----
Enabled
```

- Untuk detail API, lihat GetObjectLockConfigurationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan `GetObjectRetention` dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetObjectRetention`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kunci objek Amazon S3

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// Get the retention period for an S3 object.
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The object key.</param>
    /// <returns>The object retention details.</returns>
    public async Task<ObjectLockRetention> GetObjectRetention(string bucketName,
        string objectKey)
    {
        try
        {
            var request = new GetObjectRetentionRequest()
            {
                BucketName = bucketName,
                Key = objectKey
            };

            var response = await _amazonS3.GetObjectRetentionAsync(request);
            Console.WriteLine($"\tObject retention for {objectKey} in
{bucketName}: " +
                              $"\n\t{response.Retention.Mode} until
{response.Retention.RetainUntilDate:d}.");
            return response.Retention;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to fetch object lock retention:
'{ex.Message}'");
            return new ObjectLockRetention();
        }
    }
```

- Untuk detail API, lihat GetObjectRetentiondi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk mengambil konfigurasi retensi objek untuk objek

`get-object-retention`Contoh berikut mengambil konfigurasi retensi objek untuk objek tertentu.

```
aws s3api get-object-retention \
    --bucket my-bucket-with-object-lock \
    --key doc1.rtf
```

Output:

```
{
    "Retention": {
        "Mode": "GOVERNANCE",
        "RetainUntilDate": "2025-01-01T00:00:00.000Z"
    }
}
```

- Untuk detail API, lihat GetObjectRetentiondi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// S3Actions wraps S3 service actions.
type S3Actions struct {
```

```
 S3Client  *s3.Client
 S3Manager *manager.Uploader
}



// GetObjectRetention retrieves the object retention configuration for an S3
 object.
func (actor S3Actions) GetObjectRetention(ctx context.Context, bucket string, key
 string) (*types.ObjectLockRetention, error) {
 var retention *types.ObjectLockRetention
 input := &s3.GetObjectRetentionInput{
  Bucket: aws.String(bucket),
  Key:    aws.String(key),
 }

 output, err := actor.S3Client.GetObjectRetention(ctx, input)
 if err != nil {
  var noKey *types.NoSuchKey
  var apiErr *smithy.GenericAPIError
  if errors.As(err, &noKey) {
   log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
   err = noKey
  } else if errors.As(err, &apiErr) {
   switch apiErr.ErrorCode() {
   case "NoSuchObjectLockConfiguration":
    err = nil
   case "InvalidRequest":
    log.Printf("Bucket %s does not have locking enabled.", bucket)
    err = nil
   }
  }
 } else {
  retention = output.Retention
 }

 return retention, err
}
```

- Untuk detail API, lihat GetObjectRetentiondi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    // Get the retention period for an S3 object.
    public ObjectLockRetention getObjectRetention(String bucketName, String key){
        try {
            GetObjectRetentionRequest retentionRequest =
GetObjectRetentionRequest.builder()
                .bucket(bucketName)
                .key(key)
                .build();

            GetObjectRetentionResponse response =
getClient().getObjectRetention(retentionRequest);
            System.out.println("tObject retention for "+key +"
in "+ bucketName +": " + response.retention().mode() +" until "+
response.retention().retainUntilDate() +".");
            return response.retention();

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return null;
        }
    }
```

- Untuk detail API, lihat GetObjectRetentiondi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { fileURLToPath } from "url";
import { GetObjectRetentionCommand, S3Client } from "@aws-sdk/client-s3";

/**
 * @param {S3Client} client
 * @param {string} bucketName
 * @param {string} objectKey
 */
export const main = async (client, bucketName, objectKey) => {
  const command = new GetObjectRetentionCommand({
    Bucket: bucketName,
    Key: objectKey,
    // Optionally, you can provide additional parameters
    // ExpectedBucketOwner: "ACCOUNT_ID",
    // RequestPayer: "requester",
    // VersionId: "OBJECT_VERSION_ID",
  });

  try {
    const { Retention } = await client.send(command);
    console.log(`Object Retention Settings: ${Retention.Status}`);
  } catch (err) {
    console.error(err);
  }
};

// Invoke main function if this file was run directly.
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  main(new S3Client(), "BUCKET_NAME", "OBJECT_KEY");
}
```

- Untuk detail API, lihat GetObjectRetentiondi Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah mengembalikan mode dan tanggal sampai objek akan dipertahankan.

```
Get-S3ObjectRetention -BucketName 's3buckettesting' -Key 'testfile.txt'
```

- Untuk detail API, lihat GetObjectRetentiondi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetObjectTagging`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetObjectTagging`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai dengan tanda

CLI

AWS CLI

Untuk mengambil tag yang dilampirkan ke objek

`get-object-tagging`Contoh berikut mengambil nilai-nilai untuk kunci tertentu dari objek tertentu.

```
aws s3api get-object-tagging \
    --bucket my-bucket \
```

```
    --key doc1.rtf
```

Output:

```
{
    "TagSet": [
        {
            "Value": "confidential",
            "Key": "designation"
        }
    ]
}
```

`get-object-tagging`Contoh berikut mencoba untuk mengambil set tag objek`doc2.rtf`, yang tidak memiliki tag.

```
aws s3api get-object-tagging \
    --bucket my-bucket \
    --key doc2.rtf
```

Output:

```
{
    "TagSet": []
}
```

`get-object-tagging`Contoh berikut mengambil set tag objek`doc3.rtf`, yang memiliki beberapa tag.

```
aws s3api get-object-tagging \
    --bucket my-bucket \
    --key doc3.rtf
```

Output:

```
{
    "TagSet": [
        {
            "Value": "confidential",
```

```
            "Key": "designation"
        },
        {
            "Value": "finance",
            "Key": "department"
        },
        {
            "Value": "payroll",
            "Key": "team"
        }
    ]
}
```

- Untuk detail API, lihat GetObjectTaggingdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Sampel mengembalikan tag yang terkait dengan objek yang ada pada bucket S3 yang diberikan.

```
Get-S3ObjectTagSet -Key 'testfile.txt' -BucketName 'testbucket123'
```

Output:

```
Key  Value
---  -----
test value
```

- Untuk detail API, lihat GetObjectTaggingdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetPublicAccessBlock`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetPublicAccessBlock`.

CLI

AWS CLI

Untuk menyetel atau memodifikasi konfigurasi blokir akses publik untuk bucket

`get-public-access-block`Contoh berikut menampilkan konfigurasi blokir akses publik untuk bucket yang ditentukan.

```
aws s3api get-public-access-block \
    --bucket my-bucket
```

Output:

```
{
    "PublicAccessBlockConfiguration": {
        "IgnorePublicAcls": true,
        "BlockPublicPolicy": true,
        "BlockPublicAcls": true,
        "RestrictPublicBuckets": true
    }
}
```

- Untuk detail API, lihat GetPublicAccessBlockdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah mengembalikan konfigurasi blok akses publik dari bucket S3 yang diberikan.

```
Get-S3PublicAccessBlock -BucketName 's3testbucket'
```

- Untuk detail API, lihat GetPublicAccessBlockdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **HeadBucket** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`HeadBucket`.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
###############################################################################
# function bucket_exists
#
# This function checks to see if the specified bucket already exists.
#
# Parameters:
#       $1 - The name of the bucket to check.
#
# Returns:
#       0 - If the bucket already exists.
#       1 - If the bucket doesn't exist.
###############################################################################
function bucket_exists() {
  local bucket_name
  bucket_name=$1

  # Check whether the bucket already exists.
  # We suppress all output - we're interested only in the return code.

  if aws s3api head-bucket \
    --bucket "$bucket_name" \
    >/dev/null 2>&1; then
    return 0 # 0 in Bash script means true.
  else
    return 1 # 1 in Bash script means false.
  fi
}
```

- Untuk detail API, lihat HeadBucketdi Referensi AWS CLI Perintah.

CLI

AWS CLI

Perintah berikut memverifikasi akses ke bucket bernama`my-bucket`:

```
aws s3api head-bucket --bucket my-bucket
```

Jika bucket ada dan Anda memiliki akses ke sana, tidak ada output yang dikembalikan. Jika tidak, pesan kesalahan akan ditampilkan. Sebagai contoh:

```
A client error (404) occurred when calling the HeadBucket operation: Not Found
```

- Untuk detail API, lihat HeadBucketdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}
```

```
// BucketExists checks whether a bucket exists in the current account.
func (basics BucketBasics) BucketExists(bucketName string) (bool, error) {
 _, err := basics.S3Client.HeadBucket(context.TODO(), &s3.HeadBucketInput{
  Bucket: aws.String(bucketName),
 })
 exists := true
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NotFound:
    log.Printf("Bucket %v is available.\n", bucketName)
    exists = false
    err = nil
   default:
    log.Printf("Either you don't have access to bucket %v or another error
 occurred. "+
     "Here's what happened: %v\n", bucketName, err)
   }
  }
 } else {
  log.Printf("Bucket %v exists and you already own it.", bucketName)
 }

 return exists, err
}
```

- Untuk detail API, lihat HeadBucketdi Referensi AWS SDK for Go API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
```

```
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def exists(self):
        """
        Determine whether the bucket exists and you have access to it.

        :return: True when the bucket exists; otherwise, False.
        """
        try:
            self.bucket.meta.client.head_bucket(Bucket=self.bucket.name)
            logger.info("Bucket %s exists.", self.bucket.name)
            exists = True
        except ClientError:
            logger.warning(
                "Bucket %s doesn't exist or you don't have access to it.",
                self.bucket.name,
            )
            exists = False
        return exists
```

- Untuk detail API, lihat HeadBucketdi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`HeadObject`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`HeadObject`.

CLI

AWS CLI

Perintah berikut mengambil metadata untuk objek dalam bucket bernama: `my-bucket`

```
aws s3api head-object --bucket my-bucket --key index.html
```

Output:

```
{
    "AcceptRanges": "bytes",
    "ContentType": "text/html",
    "LastModified": "Thu, 16 Apr 2015 18:19:14 GMT",
    "ContentLength": 77,
    "VersionId": "null",
    "ETag": "\"30a6ec7e1a9ad79c203d05a589c8b400\"",
    "Metadata": {}
}
```

- Untuk detail API, lihat HeadObjectdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Tentukan jenis konten suatu objek.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.HeadObjectRequest;
import software.amazon.awssdk.services.s3.model.HeadObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;

/**
```

```
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class GetObjectContentType {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName>>

                Where:
                    bucketName - The Amazon S3 bucket name.\s
                    keyName - The key name.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        getContentType(s3, bucketName, keyName);
        s3.close();
    }

    public static void getContentType(S3Client s3, String bucketName, String
 keyName) {
        try {
            HeadObjectRequest objectRequest = HeadObjectRequest.builder()
                    .key(keyName)
                    .bucket(bucketName)
                    .build();
```

```
            HeadObjectResponse objectHead = s3.headObject(objectRequest);
            String type = objectHead.contentType();
            System.out.println("The object content type is " + type);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

Dapatkan status pemulihan suatu objek.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.HeadObjectRequest;
import software.amazon.awssdk.services.s3.model.HeadObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;

public class GetObjectRestoreStatus {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName>\s

                Where:
                    bucketName - The Amazon S3 bucket name.\s
                    keyName - A key name that represents the object.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();
```

```
        checkStatus(s3, bucketName, keyName);
        s3.close();
    }

    public static void checkStatus(S3Client s3, String bucketName, String
 keyName) {
        try {
            HeadObjectRequest headObjectRequest = HeadObjectRequest.builder()
                    .bucket(bucketName)
                    .key(keyName)
                    .build();

            HeadObjectResponse response = s3.headObject(headObjectRequest);
            System.out.println("The Amazon S3 object restoration status is " +
 response.restore());

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat HeadObjectdi Referensi AWS SDK for Java 2.x API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectExistsWrapper
  attr_reader :object
```

```
  # @param object [Aws::S3::Object] An Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Checks whether the object exists.
  #
  # @return [Boolean] True if the object exists; otherwise false.
  def exists?
    @object.exists?
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't check existence of object
 #{@object.bucket.name}:#{@object.key}. Here's why: #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object.txt"

  wrapper = ObjectExistsWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
  exists = wrapper.exists?

  puts "Object #{object_key} #{exists ? 'does' : 'does not'} exist."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk detail API, lihat HeadObjectdi Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListBucketAnalyticsConfigurations`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListBucketAnalyticsConfigurations`.

CLI

AWS CLI

Untuk mengambil daftar konfigurasi analitik untuk bucket

Berikut ini akan `list-bucket-analytics-configurations` mengambil daftar konfigurasi analitik untuk bucket yang ditentukan.

```
aws s3api list-bucket-analytics-configurations \
    --bucket my-bucket
```

Output:

```
{
    "AnalyticsConfigurationList": [
        {
            "StorageClassAnalysis": {},
            "Id": "1"
        }
    ],
    "IsTruncated": false
}
```

- Untuk detail API, lihat ListBucketAnalyticsConfigurationsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan 100 konfigurasi analitik pertama dari bucket S3 yang diberikan.

```
Get-S3BucketAnalyticsConfigurationList -BucketName 's3casetestbucket'
```

- Untuk detail API, lihat ListBucketAnalyticsConfigurationsdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListBucketInventoryConfigurations`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListBucketInventoryConfigurations`.

CLI

AWS CLI

Untuk mengambil daftar konfigurasi inventaris untuk bucket

`list-bucket-inventory-configurations`Contoh berikut mencantumkan konfigurasi inventaris untuk bucket yang ditentukan.

```
aws s3api list-bucket-inventory-configurations \
    --bucket my-bucket
```

Output:

```
{
    "InventoryConfigurationList": [
        {
            "IsEnabled": true,
            "Destination": {
                "S3BucketDestination": {
                    "Format": "ORC",
                    "Bucket": "arn:aws:s3:::my-bucket",
                    "AccountId": "123456789012"
                }
            },
            "IncludedObjectVersions": "Current",
            "Id": "1",
            "Schedule": {
                "Frequency": "Weekly"
            }
        },
        {
```

```
            "IsEnabled": true,
            "Destination": {
                "S3BucketDestination": {
                    "Format": "CSV",
                    "Bucket": "arn:aws:s3:::my-bucket",
                    "AccountId": "123456789012"
                }
            },
            "IncludedObjectVersions": "Current",
            "Id": "2",
            "Schedule": {
                "Frequency": "Daily"
            }
        }
    ],
    "IsTruncated": false
}
```

- Untuk detail API, lihat ListBucketInventoryConfigurationsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan 100 konfigurasi inventaris pertama dari bucket S3 yang diberikan.

```
Get-S3BucketInventoryConfigurationList -BucketName 's3testbucket'
```

- Untuk detail API, lihat ListBucketInventoryConfigurationsdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListBuckets`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListBuckets`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
namespace ListBucketsExample
{
    using System;
    using System.Collections.Generic;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example uses the AWS SDK for .NET to list the Amazon Simple Storage
    /// Service (Amazon S3) buckets belonging to the default account.
    /// </summary>
    public class ListBuckets
    {
        private static IAmazonS3 _s3Client;

        /// <summary>
        /// Get a list of the buckets owned by the default user.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client object.</param>
        /// <returns>The response from the ListingBuckets call that contains a
        /// list of the buckets owned by the default user.</returns>
        public static async Task<ListBucketsResponse> GetBuckets(IAmazonS3
 client)
        {
            return await client.ListBucketsAsync();
        }

        /// <summary>
        /// This method lists the name and creation date for the buckets in
        /// the passed List of S3 buckets.
        /// </summary>
```

```
        /// <param name="bucketList">A List of S3 bucket objects.</param>
        public static void DisplayBucketList(List<S3Bucket> bucketList)
        {
            bucketList
                .ForEach(b => Console.WriteLine($"Bucket name: {b.BucketName},
 created on: {b.CreationDate}"));
        }

        public static async Task Main()
        {
            // The client uses the AWS Region of the default user.
            // If the Region where the buckets were created is different,
            // pass the Region to the client constructor. For example:
            // _s3Client = new AmazonS3Client(RegionEndpoint.USEast1);
            _s3Client = new AmazonS3Client();
            var response = await GetBuckets(_s3Client);
            DisplayBucketList(response.Buckets);
        }
    }
}
```

- Untuk detail API, lihat ListBucketsdi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::listBuckets(const Aws::S3::S3ClientConfiguration &clientConfig)
 {
    Aws::S3::S3Client client(clientConfig);

    auto outcome = client.ListBuckets();

    bool result = true;
```

```
    if (!outcome.IsSuccess()) {
        std::cerr << "Failed with error: " << outcome.GetError() << std::endl;
        result = false;
    } else {
        std::cout << "Found " << outcome.GetResult().GetBuckets().size() << "
 buckets\n";
        for (auto &&b: outcome.GetResult().GetBuckets()) {
            std::cout << b.GetName() << std::endl;
        }
    }

    return result;
}
```

- Untuk detail API, lihat ListBucketsdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Perintah berikut menggunakan `list-buckets` perintah untuk menampilkan nama semua bucket Amazon S3 Anda (di semua wilayah):

```
aws s3api list-buckets --query "Buckets[].Name"
```

Opsi kueri menyaring output dari `list-buckets` down ke hanya nama bucket.

Untuk informasi selengkapnya tentang bucket, lihat Bekerja dengan Bucket Amazon S3 di Panduan Pengembang Amazon S3.

- Untuk detail API, lihat ListBucketsdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// ListBuckets lists the buckets in the current account.
func (basics BucketBasics) ListBuckets() ([]types.Bucket, error) {
 result, err := basics.S3Client.ListBuckets(context.TODO(),
 &s3.ListBucketsInput{})
 var buckets []types.Bucket
 if err != nil {
  log.Printf("Couldn't list buckets for your account. Here's why: %v\n", err)
 } else {
  buckets = result.Buckets
 }
 return buckets, err
}
```

- Untuk detail API, lihat ListBucketsdi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
```

```
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.Bucket;
import software.amazon.awssdk.services.s3.model.ListBucketsResponse;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListBuckets {
    public static void main(String[] args) {
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
            .region(region)
            .build();

        listAllBuckets(s3);

    }
    public static void listAllBuckets(S3Client s3) {
        ListBucketsResponse response = s3.listBuckets();
        List<Bucket> bucketList = response.buckets();
        for (Bucket bucket: bucketList) {
            System.out.println("Bucket name "+bucket.name());
        }
    }
}
```

- Untuk detail API, lihat ListBucketsdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat daftar bucket.

```
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new ListBucketsCommand({});

  try {
    const { Owner, Buckets } = await client.send(command);
    console.log(
      `${Owner.DisplayName} owns ${Buckets.length} bucket${
        Buckets.length === 1 ? "" : "s"
      }:`,
    );
    console.log(`${Buckets.map((b) => ` • ${b.Name}`).join("\n")}`);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat ListBucketsdi Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengembalikan semua bucket S3.

```
Get-S3Bucket
```

Contoh 2: Perintah ini mengembalikan bucket bernama “test-files”

```
Get-S3Bucket -BucketName test-files
```

- Untuk detail API, lihat ListBucketsdi Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    @staticmethod
    def list(s3_resource):
        """
        Get the buckets in all Regions for the current account.

        :param s3_resource: A Boto3 S3 resource. This is a high-level resource in
 Boto3
                            that contains collections and factory methods to
 create
```

```
                                other high-level S3 sub-resources.
        :return: The list of buckets.
        """
        try:
            buckets = list(s3_resource.buckets.all())
            logger.info("Got buckets: %s.", buckets)
        except ClientError:
            logger.exception("Couldn't get buckets.")
            raise
        else:
            return buckets
```

- Untuk detail API, lihat ListBucketsdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
require "aws-sdk-s3"

# Wraps Amazon S3 resource actions.
class BucketListWrapper
  attr_reader :s3_resource

  # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
  def initialize(s3_resource)
    @s3_resource = s3_resource
  end

  # Lists buckets for the current account.
  #
  # @param count [Integer] The maximum number of buckets to list.
  def list_buckets(count)
    puts "Found these buckets:"
```

```
    @s3_resource.buckets.each do |bucket|
      puts "\t#{bucket.name}"
      count -= 1
      break if count.zero?
    end
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't list buckets. Here's why: #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  wrapper = BucketListWrapper.new(Aws::S3::Resource.new)
  wrapper.list_buckets(25)
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk detail API, lihat ListBucketsdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
async fn show_buckets(strict: bool, client: &Client, region: &str) -> Result<(),
 Error> {
    let resp = client.list_buckets().send().await?;
    let buckets = resp.buckets();
    let num_buckets = buckets.len();

    let mut in_region = 0;

    for bucket in buckets {
```

```
        if strict {
            let r = client
                .get_bucket_location()
                .bucket(bucket.name().unwrap_or_default())
                .send()
                .await?;

            if r.location_constraint().unwrap().as_ref() == region {
                println!("{}", bucket.name().unwrap_or_default());
                in_region += 1;
            }
        } else {
            println!("{}", bucket.name().unwrap_or_default());
        }
    }

    println!();
    if strict {
        println!(
            "Found {} buckets in the {} region out of a total of {} buckets.",
            in_region, region, num_buckets
        );
    } else {
        println!("Found {} buckets in all regions.", num_buckets);
    }

    Ok(())
}
```

- Untuk detail API, lihat ListBucketsreferensi AWS SDK for Rust API.

Swift

SDK untuk Swift

> **Note**
>
> Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
/// Return an array containing information about every available bucket.
///
/// - Returns: An array of ``S3ClientTypes.Bucket`` objects describing
///   each bucket.
public func getAllBuckets() async throws -> [S3ClientTypes.Bucket] {
    let output = try await client.listBuckets(input: ListBucketsInput())

    guard let buckets = output.buckets else {
        return []
    }
    return buckets
}
```

- Untuk detail API, lihat referensi ListBuckets AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListMultipartUploads`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListMultipartUploads`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Hapus unggahan multipart yang tidak lengkap

CLI

AWS CLI

Perintah berikut mencantumkan semua unggahan multipart aktif untuk bucket bernama: `my-bucket`

```
aws s3api list-multipart-uploads --bucket my-bucket
```

Output:

```
{
    "Uploads": [
        {
            "Initiator": {
                "DisplayName": "username",
                "ID": "arn:aws:iam::0123456789012:user/username"
            },
            "Initiated": "2015-06-02T18:01:30.000Z",
            "UploadId":
 "dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3
            "StorageClass": "STANDARD",
            "Key": "multipart/01",
            "Owner": {
                "DisplayName": "aws-account-name",
                "ID":
 "100719349fc3b6dcd7c820a124bf7aecd408092c3d7b51b38494939801fc248b"
            }
        }
    ],
    "CommonPrefixes": []
}
```

Unggahan multipart yang sedang berlangsung menimbulkan biaya penyimpanan di Amazon S3. Selesaikan atau batalkan unggahan multibagian aktif untuk menghapus bagian-bagiannya dari akun Anda.

- Untuk detail API, lihat ListMultipartUploadsdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ListMultipartUploadsRequest;
import software.amazon.awssdk.services.s3.model.ListMultipartUploadsResponse;
import software.amazon.awssdk.services.s3.model.MultipartUpload;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class ListMultipartUploads {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName>\s

                Where:
                    bucketName - The name of the Amazon S3 bucket where an in-
progress multipart upload is occurring.
                """;

        if (args.length != 1) {
            System.out.println(usage);
```

```
            System.exit(1);
        }

        String bucketName = args[0];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();
        listUploads(s3, bucketName);
        s3.close();
    }

    public static void listUploads(S3Client s3, String bucketName) {
        try {
            ListMultipartUploadsRequest listMultipartUploadsRequest =
 ListMultipartUploadsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListMultipartUploadsResponse response =
 s3.listMultipartUploads(listMultipartUploadsRequest);
            List<MultipartUpload> uploads = response.uploads();
            for (MultipartUpload upload : uploads) {
                System.out.println("Upload in progress: Key = \"" + upload.key()
 + "\", id = " + upload.uploadId());
            }

        } catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat ListMultipartUploadsdi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListObjectVersions`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListObjectVersions`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Bekerja dengan objek berversi

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
using System;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example lists the versions of the objects in a version enabled
/// Amazon Simple Storage Service (Amazon S3) bucket.
/// </summary>
public class ListObjectVersions
{
    public static async Task Main()
    {
        string bucketName = "doc-example-bucket";

        // If the AWS Region where your bucket is defined is different from
        // the AWS Region where the Amazon S3 bucket is defined, pass the
constant
        // for the AWS Region to the client constructor like this:
        //      var client = new AmazonS3Client(RegionEndpoint.USWest2);
        IAmazonS3 client = new AmazonS3Client();
        await GetObjectListWithAllVersionsAsync(client, bucketName);
```

```
        }

        /// <summary>
        /// This method lists all versions of the objects within an Amazon S3
        /// version enabled bucket.
        /// </summary>
        /// <param name="client">The initialized client object used to call
        /// ListVersionsAsync.</param>
        /// <param name="bucketName">The name of the version enabled Amazon S3
 bucket
        /// for which you want to list the versions of the contained objects.</
param>
        public static async Task GetObjectListWithAllVersionsAsync(IAmazonS3
 client, string bucketName)
        {
            try
            {
                // When you instantiate the ListVersionRequest, you can
                // optionally specify a key name prefix in the request
                // if you want a list of object versions of a specific object.

                // For this example we set a small limit in MaxKeys to return
                // a small list of versions.
                ListVersionsRequest request = new ListVersionsRequest()
                {
                    BucketName = bucketName,
                    MaxKeys = 2,
                };

                do
                {
                    ListVersionsResponse response = await
 client.ListVersionsAsync(request);

                    // Process response.
                    foreach (S3ObjectVersion entry in response.Versions)
                    {
                        Console.WriteLine($"key: {entry.Key} size:
 {entry.Size}");
                    }

                    // If response is truncated, set the marker to get the next
                    // set of keys.
                    if (response.IsTruncated)
```

```
                    {
                        request.KeyMarker = response.NextKeyMarker;
                        request.VersionIdMarker = response.NextVersionIdMarker;
                    }
                    else
                    {
                        request = null;
                    }
                }
                while (request != null);
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error: '{ex.Message}'");
            }
        }
    }
```

- Untuk detail API, lihat ListObjectVersionsdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Perintah berikut mengambil informasi versi untuk objek dalam bucket bernama`my-bucket`:

```
aws s3api list-object-versions --bucket my-bucket --prefix index.html
```

Output:

```
{
    "DeleteMarkers": [
        {
            "Owner": {
                "DisplayName": "my-username",
                "ID":
 "7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": true,
            "VersionId": "B2VsEK5saUNNHKcOAJj7hIE86RozToyq",
```

```
            "Key": "index.html",
            "LastModified": "2015-11-10T00:57:03.000Z"
        },
        {
            "Owner": {
                "DisplayName": "my-username",
                "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": false,
            "VersionId": ".FLQEZscLIcfxSq.jsFJ.szUkmng2Yw6",
            "Key": "index.html",
            "LastModified": "2015-11-09T23:32:20.000Z"
        }
    ],
    "Versions": [
        {
            "LastModified": "2015-11-10T00:20:11.000Z",
            "VersionId": "Rb_l2T8UHDkFEwCgJjhlgPOZC0qJ.vpD",
            "ETag": "\"0622528de826c0df5db1258a23b80be5\"",
            "StorageClass": "STANDARD",
            "Key": "index.html",
            "Owner": {
                "DisplayName": "my-username",
                "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": false,
            "Size": 38
        },
        {
            "LastModified": "2015-11-09T23:26:41.000Z",
            "VersionId": "rasWWGpgk9E4s0LyTJgusGeRQKLVIAFf",
            "ETag": "\"06225825b8028de826c0df5db1a23be5\"",
            "StorageClass": "STANDARD",
            "Key": "index.html",
            "Owner": {
                "DisplayName": "my-username",
                "ID":
"7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": false,
            "Size": 38
        },
```

```
        {
            "LastModified": "2015-11-09T22:50:50.000Z",
            "VersionId": "null",
            "ETag": "\"d1f45267a863c8392e07d24dd592f1b9\"",
            "StorageClass": "STANDARD",
            "Key": "index.html",
            "Owner": {
                "DisplayName": "my-username",
                "ID":
 "7009a8971cd660687538875e7c86c5b672fe116bd438f46db45460ddcd036c32"
            },
            "IsLatest": false,
            "Size": 533823
        }
    ]
}
```

- Untuk detail API, lihat ListObjectVersionsdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// S3Actions wraps S3 service actions.
type S3Actions struct {
 S3Client  *s3.Client
 S3Manager *manager.Uploader
}



// ListObjectVersions lists all versions of all objects in a bucket.
func (actor S3Actions) ListObjectVersions(ctx context.Context, bucket string)
 ([]types.ObjectVersion, error) {
 var err error
```

```
 var output *s3.ListObjectVersionsOutput
 var versions []types.ObjectVersion
 input := &s3.ListObjectVersionsInput{Bucket: aws.String(bucket)}
 versionPaginator := s3.NewListObjectVersionsPaginator(actor.S3Client, input)
 for versionPaginator.HasMorePages() {
  output, err = versionPaginator.NextPage(ctx)
  if err != nil {
   var noBucket *types.NoSuchBucket
   if errors.As(err, &noBucket) {
    log.Printf("Bucket %s does not exist.\n", bucket)
    err = noBucket
   }
   break
  } else {
   versions = append(versions, output.Versions...)
  }
 }
 return versions, err
}
```

- Untuk detail API, lihat ListObjectVersionsdi Referensi AWS SDK for Go API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
async fn show_versions(client: &Client, bucket: &str) -> Result<(), Error> {
    let resp = client.list_object_versions().bucket(bucket).send().await?;

    for version in resp.versions() {
        println!("{}", version.key().unwrap_or_default());
        println!("  version ID: {}", version.version_id().unwrap_or_default());
        println!();
    }
```

```
    Ok(())
}
```

- Untuk detail API, lihat ListObjectVersionsreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListObjects`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListObjects`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Membuat halaman web yang mencantumkan objek Amazon S3

CLI

AWS CLI

Contoh berikut menggunakan `list-objects` perintah untuk menampilkan nama-nama semua objek dalam bucket yang ditentukan:

```
aws s3api list-objects --bucket text-content --query 'Contents[].{Key: Key, Size:
 Size}'
```

Contoh menggunakan `--query` argumen untuk memfilter output `list-objects` turun ke nilai kunci dan ukuran untuk setiap objek

Untuk informasi selengkapnya tentang objek, lihat Bekerja dengan Objek Amazon S3 di Panduan Pengembang Amazon S3.

- Untuk detail API, lihat ListObjectsdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengambil informasi tentang semua item di bucket “test-files”.

```
Get-S3Object -BucketName test-files
```

Contoh 2: Perintah ini mengambil informasi tentang item "sample.txt" dari bucket “test-files”.

```
Get-S3Object -BucketName test-files -Key sample.txt
```

Contoh 3: Perintah ini mengambil informasi tentang semua item dengan awalan “sample” dari bucket “test-files”.

```
Get-S3Object -BucketName test-files -KeyPrefix sample
```

- Untuk detail API, lihat ListObjectsdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListObjectsV2`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListObjectsV2`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai bucket dan objek

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        /// <summary>
        /// Shows how to list the objects in an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client object.</param>
        /// <param name="bucketName">The name of the bucket for which to list
        /// the contents.</param>
        /// <returns>A boolean value indicating the success or failure of the
        /// copy operation.</returns>
        public static async Task<bool> ListBucketContentsAsync(IAmazonS3 client,
string bucketName)
        {
            try
            {
                var request = new ListObjectsV2Request
                {
                    BucketName = bucketName,
                    MaxKeys = 5,
                };

                Console.WriteLine("--------------------------------------");
                Console.WriteLine($"Listing the contents of {bucketName}:");
                Console.WriteLine("--------------------------------------");

                ListObjectsV2Response response;

                do
                {
                    response = await client.ListObjectsV2Async(request);

                    response.S3Objects
```

```
                        .ForEach(obj => Console.WriteLine($"{obj.Key,-35}
{obj.LastModified.ToShortDateString(),10}{obj.Size,10}"));

                    // If the response is truncated, set the request
 ContinuationToken
                    // from the NextContinuationToken property of the response.
                    request.ContinuationToken = response.NextContinuationToken;
                }
                while (response.IsTruncated);

                return true;
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error encountered on server.
 Message:'{ex.Message}' getting list of objects.");
                return false;
            }
        }
```

Daftar objek dengan paginator.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// The following example lists objects in an Amazon Simple Storage
    /// Service (Amazon S3) bucket.
    /// </summary>
    public class ListObjectsPaginator
    {
        private const string BucketName = "doc-example-bucket";

        public static async Task Main()
        {
            IAmazonS3 s3Client = new AmazonS3Client();

            Console.WriteLine($"Listing the objects contained in {BucketName}:
\n");
```

```
            await ListingObjectsAsync(s3Client, BucketName);
        }

        /// <summary>
        /// This method uses a paginator to retrieve the list of objects in an
        /// an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">An Amazon S3 client object.</param>
        /// <param name="bucketName">The name of the S3 bucket whose objects
        /// you want to list.</param>
        public static async Task ListingObjectsAsync(IAmazonS3 client, string
bucketName)
        {
            var listObjectsV2Paginator = client.Paginators.ListObjectsV2(new
ListObjectsV2Request
            {
                BucketName = bucketName,
            });

            await foreach (var response in listObjectsV2Paginator.Responses)
            {
                Console.WriteLine($"HttpStatusCode: {response.HttpStatusCode}");
                Console.WriteLine($"Number of Keys: {response.KeyCount}");
                foreach (var entry in response.S3Objects)
                {
                    Console.WriteLine($"Key = {entry.Key} Size = {entry.Size}");
                }
            }
        }
    }
```

- Untuk detail API, lihat ListObjectsV2 di Referensi AWS SDK for .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function list_items_in_bucket
#
# This function displays a list of the files in the bucket with each file's
# size. The function uses the --query parameter to retrieve only the key and
# size fields from the Contents collection.
#
# Parameters:
#       $1 - The name of the bucket.
#
# Returns:
#       The list of files in text format.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function list_items_in_bucket() {
  local bucket_name=$1
  local response

  response=$(aws s3api list-objects \
    --bucket "$bucket_name" \
    --output text \
```

```
    --query 'Contents[].{Key: Key, Size: Size}')

  # shellcheck disable=SC2181
  if [[ ${?} -eq 0 ]]; then
    echo "$response"
  else
    errecho "ERROR: AWS reports s3api list-objects operation failed.\n$response"
    return 1
  fi
}
```

- Untuk detail API, lihat ListObjectsV2 di Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::listObjects(const Aws::String &bucketName,
                             const Aws::S3::S3ClientConfiguration &clientConfig)
 {
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::ListObjectsV2Request request;
    request.WithBucket(bucketName);

    Aws::String continuationToken; // Used for pagination.
    Aws::Vector<Aws::S3::Model::Object> allObjects;

    do {
        if (!continuationToken.empty()) {
            request.SetContinuationToken(continuationToken);
        }

        auto outcome = s3Client.ListObjectsV2(request);
```

```
        if (!outcome.IsSuccess()) {
            std::cerr << "Error: listObjects: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        } else {
            Aws::Vector<Aws::S3::Model::Object> objects =
                    outcome.GetResult().GetContents();

            allObjects.insert(allObjects.end(), objects.begin(), objects.end());
            continuationToken = outcome.GetResult().GetNextContinuationToken();
        }
    } while (!continuationToken.empty());

    std::cout << allObjects.size() << " object(s) found:" << std::endl;

    for (const auto &object: allObjects) {
        std::cout << "  " << object.GetKey() << std::endl;
    }

    return true;
}
```

- Untuk detail API, lihat ListObjectsV2 di Referensi AWS SDK for C++ API.

CLI

AWS CLI

Untuk mendapatkan daftar objek dalam ember

`list-objects-v2`Contoh berikut mencantumkan objek dalam bucket yang ditentukan.

```
aws s3api list-objects-v2 \
    --bucket my-bucket
```

Output:

```
{
    "Contents": [
        {
            "LastModified": "2019-11-05T23:11:50.000Z",
            "ETag": "\"621503c373607d548b37cff8778d992c\"",
```

```
            "StorageClass": "STANDARD",
            "Key": "doc1.rtf",
            "Size": 391
        },
        {
            "LastModified": "2019-11-05T23:11:50.000Z",
            "ETag": "\"a2cecc36ab7c7fe3a71a273b9d45b1b5\"",
            "StorageClass": "STANDARD",
            "Key": "doc2.rtf",
            "Size": 373
        },
        {
            "LastModified": "2019-11-05T23:11:50.000Z",
            "ETag": "\"08210852f65a2e9cb999972539a64d68\"",
            "StorageClass": "STANDARD",
            "Key": "doc3.rtf",
            "Size": 399
        },
        {
            "LastModified": "2019-11-05T23:11:50.000Z",
            "ETag": "\"d1852dd683f404306569471af106988e\"",
            "StorageClass": "STANDARD",
            "Key": "doc4.rtf",
            "Size": 6225
        }
    ]
}
```

- Untuk detail API, lihat ListObjectsV2 di Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// ListObjects lists the objects in a bucket.
func (basics BucketBasics) ListObjects(bucketName string) ([]types.Object, error)
 {
 result, err := basics.S3Client.ListObjectsV2(context.TODO(),
 &s3.ListObjectsV2Input{
  Bucket: aws.String(bucketName),
 })
 var contents []types.Object
 if err != nil {
  log.Printf("Couldn't list objects in bucket %v. Here's why: %v\n", bucketName,
 err)
 } else {
  contents = result.Contents
 }
 return contents, err
}
```

- Untuk detail API, lihat ListObjectsV2 di Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ListObjectsRequest;
import software.amazon.awssdk.services.s3.model.ListObjectsResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.S3Object;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class ListObjects {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName>\s

                Where:
                    bucketName - The Amazon S3 bucket from which objects are
 read.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        listBucketObjects(s3, bucketName);
        s3.close();
```

```
    }

    public static void listBucketObjects(S3Client s3, String bucketName) {
        try {
            ListObjectsRequest listObjects = ListObjectsRequest
                    .builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.print("\n The name of the key is " + myValue.key());
                System.out.print("\n The object is " + calKb(myValue.size()) + "
 KBs");
                System.out.print("\n The owner is " + myValue.owner());
            }

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // convert bytes to kbs.
    private static long calKb(Long val) {
        return val / 1024;
    }
}
```

Buat daftar objek menggunakan penomoran halaman.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.paginators.ListObjectsV2Iterable;

public class ListObjectsPaginated {
    public static void main(String[] args) {
        final String usage = """
```

```
                Usage:
                    <bucketName>\s

                Where:
                    bucketName - The Amazon S3 bucket from which objects are
 read.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        listBucketObjects(s3, bucketName);
        s3.close();
    }

    public static void listBucketObjects(S3Client s3, String bucketName) {
        try {
            ListObjectsV2Request listReq = ListObjectsV2Request.builder()
                    .bucket(bucketName)
                    .maxKeys(1)
                    .build();

            ListObjectsV2Iterable listRes = s3.listObjectsV2Paginator(listReq);
            listRes.stream()
                    .flatMap(r -> r.contents().stream())
                    .forEach(content -> System.out.println(" Key: " +
 content.key() + " size = " + content.size()));

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat ListObjectsV2 di Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat daftar semua objek di bucket Anda. Jika ada lebih dari satu objek, IsTruncated dan NextContinuationToken akan digunakan untuk iterasi atas daftar lengkap.

```
import {
  S3Client,
  // This command supersedes the ListObjectsCommand and is the recommended way to
 list objects.
  ListObjectsV2Command,
} from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new ListObjectsV2Command({
    Bucket: "my-bucket",
    // The default and maximum number of keys returned is 1000. This limits it to
    // one for demonstration purposes.
    MaxKeys: 1,
  });

  try {
    let isTruncated = true;

    console.log("Your bucket contains the following objects:\n");
    let contents = "";

    while (isTruncated) {
      const { Contents, IsTruncated, NextContinuationToken } =
        await client.send(command);
      const contentsList = Contents.map((c) => ` • ${c.Key}`).join("\n");
```

```
      contents += contentsList + "\n";
      isTruncated = IsTruncated;
      command.input.ContinuationToken = NextContinuationToken;
    }
    console.log(contents);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk detail API, lihat ListObjectsV2 di Referensi AWS SDK for JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun listBucketObjects(bucketName: String) {
    val request =
        ListObjectsRequest {
            bucket = bucketName
        }

    S3Client { region = "us-east-1" }.use { s3 ->
        val response = s3.listObjects(request)
        response.contents?.forEach { myObject ->
            println("The name of the key is ${myObject.key}")
            println("The object is ${myObject.size?.let { calKb(it) }} KBs")
            println("The owner is ${myObject.owner}")
        }
    }
}

private fun calKb(intValue: Long): Long = intValue / 1024
```

- Untuk detail API, lihat ListObjectsV2 di AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat daftar objek dalam bucket.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        try {
            $contents = $this->s3client->listObjectsV2([
                'Bucket' => $this->bucketName,
            ]);
            echo "The contents of your bucket are: \n";
            foreach ($contents['Contents'] as $content) {
                echo $content['Key'] . "\n";
            }
        } catch (Exception $exception) {
            echo "Failed to list objects in $this->bucketName with error: " .
$exception->getMessage();
            exit("Please fix error with listing objects before continuing.");
        }
```

- Untuk detail API, lihat ListObjectsV2 di Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    @staticmethod
    def list(bucket, prefix=None):
        """
        Lists the objects in a bucket, optionally filtered by a prefix.

        :param bucket: The bucket to query. This is a Boto3 Bucket resource.
        :param prefix: When specified, only objects that start with this prefix
 are listed.
        :return: The list of objects.
        """
        try:
            if not prefix:
                objects = list(bucket.objects.all())
            else:
                objects = list(bucket.objects.filter(Prefix=prefix))
            logger.info(
                "Got objects %s from bucket '%s'", [o.key for o in objects],
 bucket.name
            )
```

```
        except ClientError:
            logger.exception("Couldn't get objects for bucket '%s'.",
 bucket.name)
            raise
        else:
            return objects
```

- Untuk detail API, lihat ListObjectsV2 di AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket actions.
class BucketListObjectsWrapper
  attr_reader :bucket

  # @param bucket [Aws::S3::Bucket] An existing Amazon S3 bucket.
  def initialize(bucket)
    @bucket = bucket
  end

  # Lists object in a bucket.
  #
  # @param max_objects [Integer] The maximum number of objects to list.
  # @return [Integer] The number of objects listed.
  def list_objects(max_objects)
    count = 0
    puts "The objects in #{@bucket.name} are:"
    @bucket.objects.each do |obj|
      puts "\t#{obj.key}"
      count += 1
```

```
      break if count == max_objects
    end
    count
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't list objects in bucket #{bucket.name}. Here's why:
 #{e.message}"
    0
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"

  wrapper = BucketListObjectsWrapper.new(Aws::S3::Bucket.new(bucket_name))
  count = wrapper.list_objects(25)
  puts "Listed #{count} objects."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk detail API, lihat ListObjectsV2 di Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
pub async fn list_objects(client: &Client, bucket: &str) -> Result<(), Error> {
    let mut response = client
        .list_objects_v2()
        .bucket(bucket.to_owned())
        .max_keys(10) // In this example, go 10 at a time.
        .into_paginator()
        .send();
```

```
    while let Some(result) = response.next().await {
        match result {
            Ok(output) => {
                for object in output.contents() {
                    println!(" - {}", object.key().unwrap_or("Unknown"));
                }
            }
            Err(err) => {
                eprintln!("{err:?}")
            }
        }
    }

    Ok(())
}
```

- Untuk detail API, lihat ListObjectsV2 di AWS SDK untuk referensi Rust API.

SAP ABAP

SDK untuk SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    TRY.
        oo_result = lo_s3->listobjectsv2(         " oo_result is returned for
testing purposes. "
          iv_bucket = iv_bucket_name
        ).
        MESSAGE 'Retrieved list of objects in S3 bucket.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat ListObjectsV2 di AWS SDK untuk referensi SAP ABAP API.

Swift

SDK untuk Swift

> **Note**
>
> Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
public func listBucketFiles(bucket: String) async throws -> [String] {
    let input = ListObjectsV2Input(
        bucket: bucket
    )
    let output = try await client.listObjectsV2(input: input)
    var names: [String] = []

    guard let objList = output.contents else {
        return []
    }

    for obj in objList {
        if let objName = obj.key {
            names.append(objName)
        }
    }

    return names
}
```

- Untuk detail API, lihat ListObjectsV2 di AWS SDK untuk referensi Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutBucketAccelerateConfiguration`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketAccelerateConfiguration`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
using System;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// Amazon Simple Storage Service (Amazon S3) Transfer Acceleration is a
/// bucket-level feature that enables you to perform faster data transfers
/// to Amazon S3. This example shows how to configure Transfer
/// Acceleration.
/// </summary>
public class TransferAcceleration
{
    /// <summary>
    /// The main method initializes the client object and sets the
    /// Amazon Simple Storage Service (Amazon S3) bucket name before
    /// calling EnableAccelerationAsync.
    /// </summary>
    public static async Task Main()
    {
        var s3Client = new AmazonS3Client();
        const string bucketName = "doc-example-bucket";
```

```
            await EnableAccelerationAsync(s3Client, bucketName);
        }

        /// <summary>
        /// This method sets the configuration to enable transfer acceleration
        /// for the bucket referred to in the bucketName parameter.
        /// </summary>
        /// <param name="client">An Amazon S3 client used to enable the
        /// acceleration on an Amazon S3 bucket.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket for which
the
        /// method will be enabling acceleration.</param>
        private static async Task EnableAccelerationAsync(AmazonS3Client client,
string bucketName)
        {
            try
            {
                var putRequest = new PutBucketAccelerateConfigurationRequest
                {
                    BucketName = bucketName,
                    AccelerateConfiguration = new AccelerateConfiguration
                    {
                        Status = BucketAccelerateStatus.Enabled,
                    },
                };
                await client.PutBucketAccelerateConfigurationAsync(putRequest);

                var getRequest = new GetBucketAccelerateConfigurationRequest
                {
                    BucketName = bucketName,
                };
                var response = await
client.GetBucketAccelerateConfigurationAsync(getRequest);

                Console.WriteLine($"Acceleration state = '{response.Status}' ");
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error occurred. Message:'{ex.Message}' when
setting transfer acceleration");
            }
        }
    }
```

- Untuk detail API, lihat PutBucketAccelerateConfigurationdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk mengatur konfigurasi percepatan ember

`put-bucket-accelerate-configuration`Contoh berikut memungkinkan konfigurasi percepatan untuk bucket yang ditentukan.

```
aws s3api put-bucket-accelerate-configuration \
    --bucket my-bucket \
    --accelerate-configuration Status=Enabled
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat PutBucketAccelerateConfigurationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini memungkinkan percepatan transfer untuk bucket S3 yang diberikan.

```
$statusVal = New-Object Amazon.S3.BucketAccelerateStatus('Enabled')
Write-S3BucketAccelerateConfiguration -BucketName 's3testbucket' -
AccelerateConfiguration_Status $statusVal
```

- Untuk detail API, lihat PutBucketAccelerateConfigurationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **PutBucketAcl** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketAcl`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Mengelola daftar kontrol akses (ACL)

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        /// <summary>
        /// Creates an Amazon S3 bucket with an ACL to control access to the
        /// bucket and the objects stored in it.
        /// </summary>
        /// <param name="client">The initialized client object used to create
        /// an Amazon S3 bucket, with an ACL applied to the bucket.
        /// </param>
        /// <param name="region">The AWS Region where the bucket will be
created.</param>
        /// <param name="newBucketName">The name of the bucket to create.</param>
        /// <returns>A boolean value indicating success or failure.</returns>
        public static async Task<bool> CreateBucketUseCannedACLAsync(IAmazonS3
client, S3Region region, string newBucketName)
        {
            try
            {
                // Create a new Amazon S3 bucket with Canned ACL.
                var putBucketRequest = new PutBucketRequest()
                {
                    BucketName = newBucketName,
                    BucketRegion = region,
```

```
                    CannedACL = S3CannedACL.LogDeliveryWrite,
                };

                PutBucketResponse putBucketResponse = await
 client.PutBucketAsync(putBucketRequest);

                return putBucketResponse.HttpStatusCode ==
 System.Net.HttpStatusCode.OK;
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Amazon S3 error: {ex.Message}");
            }

            return false;
        }
```

- Untuk detail API, lihat PutBucketAcldi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::putBucketAcl(const Aws::String &bucketName, const Aws::String
 &ownerID,
                              const Aws::String &granteePermission,
                              const Aws::String &granteeType, const Aws::String
 &granteeID,
                              const Aws::String &granteeEmailAddress,
                              const Aws::String &granteeURI, const
 Aws::S3::S3ClientConfiguration &clientConfig) {
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::Owner owner;
```

```
    owner.SetID(ownerID);

    Aws::S3::Model::Grantee grantee;
    grantee.SetType(setGranteeType(granteeType));

    if (!granteeEmailAddress.empty()) {
        grantee.SetEmailAddress(granteeEmailAddress);
    }

    if (!granteeID.empty()) {
        grantee.SetID(granteeID);
    }

    if (!granteeURI.empty()) {
        grantee.SetURI(granteeURI);
    }

    Aws::S3::Model::Grant grant;
    grant.SetGrantee(grantee);
    grant.SetPermission(setGranteePermission(granteePermission));

    Aws::Vector<Aws::S3::Model::Grant> grants;
    grants.push_back(grant);

    Aws::S3::Model::AccessControlPolicy acp;
    acp.SetOwner(owner);
    acp.SetGrants(grants);

    Aws::S3::Model::PutBucketAclRequest request;
    request.SetAccessControlPolicy(acp);
    request.SetBucket(bucketName);

    Aws::S3::Model::PutBucketAclOutcome outcome =
            s3Client.PutBucketAcl(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &error = outcome.GetError();

        std::cerr << "Error: putBucketAcl: " << error.GetExceptionName()
                  << " - " << error.GetMessage() << std::endl;
    } else {
        std::cout << "Successfully added an ACL to the bucket '" << bucketName
                  << "'." << std::endl;
    }
```

```
    return outcome.IsSuccess();
}

//! Routine which converts a human-readable string to a built-in type
 enumeration.
/*!
 \param access: Human readable string.
 \return Permission: A Permission enum.
*/

Aws::S3::Model::Permission setGranteePermission(const Aws::String &access) {
    if (access == "FULL_CONTROL")
        return Aws::S3::Model::Permission::FULL_CONTROL;
    if (access == "WRITE")
        return Aws::S3::Model::Permission::WRITE;
    if (access == "READ")
        return Aws::S3::Model::Permission::READ;
    if (access == "WRITE_ACP")
        return Aws::S3::Model::Permission::WRITE_ACP;
    if (access == "READ_ACP")
        return Aws::S3::Model::Permission::READ_ACP;
    return Aws::S3::Model::Permission::NOT_SET;
}

//! Routine which converts a human-readable string to a built-in type
 enumeration.
/*!
 \param type: Human readable string.
 \return Type: Type enumeration
*/

Aws::S3::Model::Type setGranteeType(const Aws::String &type) {
    if (type == "Amazon customer by email")
        return Aws::S3::Model::Type::AmazonCustomerByEmail;
    if (type == "Canonical user")
        return Aws::S3::Model::Type::CanonicalUser;
    if (type == "Group")
        return Aws::S3::Model::Type::Group;
    return Aws::S3::Model::Type::NOT_SET;
}
```

- Untuk detail API, lihat PutBucketAcldi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Contoh ini memberikan `full control` kepada dua AWS pengguna (user1@example.com dan user2@example.com) dan `read` izin untuk semua orang:

```
aws s3api put-bucket-acl --bucket MyBucket --grant-full-control
 emailaddress=user1@example.com,emailaddress=user2@example.com --grant-read
 uri=http://acs.amazonaws.com/groups/global/AllUsers
```

Lihat http://docs.aws.amazon.com/AmazonS3/latest/API/RESTBucketPUTacl.html untuk detail tentang ACL kustom (perintah s3api ACL, sepertiput`-bucket-acl`, menggunakan notasi argumen singkatan yang sama).

- Untuk detail API, lihat PutBucketAcldi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.AccessControlPolicy;
import software.amazon.awssdk.services.s3.model.Grant;
import software.amazon.awssdk.services.s3.model.Permission;
import software.amazon.awssdk.services.s3.model.PutBucketAclRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.Type;

import java.util.ArrayList;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
```

```
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class SetAcl {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                  <bucketName> <id>\s

                Where:
                  bucketName - The Amazon S3 bucket to grant permissions on.\s
                  id - The ID of the owner of this bucket (you can get this value
 from the AWS Management Console).
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String id = args[1];
        System.out.format("Setting access \n");
        System.out.println(" in bucket: " + bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        setBucketAcl(s3, bucketName, id);
        System.out.println("Done!");
        s3.close();
    }

    public static void setBucketAcl(S3Client s3, String bucketName, String id) {
        try {
            Grant ownerGrant = Grant.builder()
                    .grantee(builder -> builder.id(id)
                            .type(Type.CANONICAL_USER))
```

```
                    .permission(Permission.FULL_CONTROL)
                    .build();

            List<Grant> grantList2 = new ArrayList<>();
            grantList2.add(ownerGrant);

            AccessControlPolicy acl = AccessControlPolicy.builder()
                    .owner(builder -> builder.id(id))
                    .grants(grantList2)
                    .build();

            PutBucketAclRequest putAclReq = PutBucketAclRequest.builder()
                    .bucket(bucketName)
                    .accessControlPolicy(acl)
                    .build();

            s3.putBucketAcl(putAclReq);

        } catch (S3Exception e) {
            e.printStackTrace();
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat PutBucketAcldi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Tempatkan bucket ACL.

```
import { PutBucketAclCommand, S3Client } from "@aws-sdk/client-s3";
```

```
const client = new S3Client({});

// Most Amazon S3 use cases don't require the use of access control lists (ACLs).
// We recommend that you disable ACLs, except in unusual circumstances where
// you need to control access for each object individually.
// Consider a policy instead. For more information see https://
docs.aws.amazon.com/AmazonS3/latest/userguide/bucket-policies.html.
export const main = async () => {
  // Grant a user READ access to a bucket.
  const command = new PutBucketAclCommand({
    Bucket: "test-bucket",
    AccessControlPolicy: {
      Grants: [
        {
          Grantee: {
            // The canonical ID of the user. This ID is an obfuscated form of
 your AWS account number.
            // It's unique to Amazon S3 and can't be found elsewhere.
            // For more information, see https://docs.aws.amazon.com/AmazonS3/
latest/userguide/finding-canonical-user-id.html.
            ID: "canonical-id-1",
            Type: "CanonicalUser",
          },
          // One of FULL_CONTROL | READ | WRITE | READ_ACP | WRITE_ACP
          // https://docs.aws.amazon.com/AmazonS3/latest/API/
API_Grant.html#AmazonS3-Type-Grant-Permission
          Permission: "FULL_CONTROL",
        },
      ],
      Owner: {
        ID: "canonical-id-2",
      },
    },
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat PutBucketAcldi Referensi AWS SDK for JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun setBucketAcl(
    bucketName: String,
    idVal: String,
) {
    val myGrant =
        Grantee {
            id = idVal
            type = Type.CanonicalUser
        }

    val ownerGrant =
        Grant {
            grantee = myGrant
            permission = Permission.FullControl
        }

    val grantList = mutableListOf<Grant>()
    grantList.add(ownerGrant)

    val ownerOb =
        Owner {
            id = idVal
        }

    val acl =
        AccessControlPolicy {
            owner = ownerOb
            grants = grantList
```

```
        }

    val request =
        PutBucketAclRequest {
            bucket = bucketName
            accessControlPolicy = acl
        }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.putBucketAcl(request)
        println("An ACL was successfully set on $bucketName")
    }
}
```

- Untuk detail API, lihat PutBucketAcldi AWS SDK untuk referensi API Kotlin.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def grant_log_delivery_access(self):
        """
```

```
        Grant the AWS Log Delivery group write access to the bucket so that
        Amazon S3 can deliver access logs to the bucket. This is the only
recommended
        use of an S3 bucket ACL.
        """
        try:
            acl = self.bucket.Acl()
            # Putting an ACL overwrites the existing ACL. If you want to preserve
            # existing grants, append new grants to the list of existing grants.
            grants = acl.grants if acl.grants else []
            grants.append(
                {
                    "Grantee": {
                        "Type": "Group",
                        "URI": "http://acs.amazonaws.com/groups/s3/LogDelivery",
                    },
                    "Permission": "WRITE",
                }
            )
            acl.put(AccessControlPolicy={"Grants": grants, "Owner": acl.owner})
            logger.info("Granted log delivery access to bucket '%s'",
self.bucket.name)
        except ClientError:
            logger.exception("Couldn't add ACL to bucket '%s'.",
self.bucket.name)
            raise
```

- Untuk detail API, lihat PutBucketAcldi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutBucketCors`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketCors`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        /// <summary>
        /// Add CORS configuration to the Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used
        /// to apply the CORS configuration to an Amazon S3 bucket.</param>
        /// <param name="configuration">The CORS configuration to apply.</param>
        private static async Task PutCORSConfigurationAsync(AmazonS3Client
client, CORSConfiguration configuration)
        {
            PutCORSConfigurationRequest request = new
PutCORSConfigurationRequest()
            {
                BucketName = BucketName,
                Configuration = configuration,
            };

            _ = await client.PutCORSConfigurationAsync(request);
        }
```

- Untuk detail API, lihat PutBucketCorsdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Contoh berikut memungkinkan`PUT`,`POST`, dan `DELETE` permintaan dari www.example.com, dan memungkinkan `GET` permintaan dari domain apa pun:

```
aws s3api put-bucket-cors --bucket MyBucket --cors-configuration file://cors.json

cors.json:
{
  "CORSRules": [
    {
      "AllowedOrigins": ["http://www.example.com"],
      "AllowedHeaders": ["*"],
      "AllowedMethods": ["PUT", "POST", "DELETE"],
      "MaxAgeSeconds": 3000,
      "ExposeHeaders": ["x-amz-server-side-encryption"]
    },
    {
      "AllowedOrigins": ["*"],
      "AllowedHeaders": ["Authorization"],
      "AllowedMethods": ["GET"],
      "MaxAgeSeconds": 3000
    }
  ]
}
```

- Untuk detail API, lihat PutBucketCorsdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import java.util.ArrayList;
import java.util.List;
import software.amazon.awssdk.services.s3.model.GetBucketCorsRequest;
import software.amazon.awssdk.services.s3.model.GetBucketCorsResponse;
import software.amazon.awssdk.services.s3.model.DeleteBucketCorsRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
```

```
import software.amazon.awssdk.services.s3.model.CORSRule;
import software.amazon.awssdk.services.s3.model.CORSConfiguration;
import software.amazon.awssdk.services.s3.model.PutBucketCorsRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class S3Cors {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <accountId>\s

                Where:
                    bucketName - The Amazon S3 bucket to upload an object into.
                    accountId - The id of the account that owns the Amazon S3
 bucket.
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String accountId = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        setCorsInformation(s3, bucketName, accountId);
        getBucketCorsInformation(s3, bucketName, accountId);
        deleteBucketCorsInformation(s3, bucketName, accountId);
        s3.close();
    }
```

```
   public static void deleteBucketCorsInformation(S3Client s3, String
bucketName, String accountId) {
       try {
           DeleteBucketCorsRequest bucketCorsRequest =
DeleteBucketCorsRequest.builder()
                   .bucket(bucketName)
                   .expectedBucketOwner(accountId)
                   .build();

           s3.deleteBucketCors(bucketCorsRequest);

       } catch (S3Exception e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   public static void getBucketCorsInformation(S3Client s3, String bucketName,
String accountId) {
       try {
           GetBucketCorsRequest bucketCorsRequest =
GetBucketCorsRequest.builder()
                   .bucket(bucketName)
                   .expectedBucketOwner(accountId)
                   .build();

           GetBucketCorsResponse corsResponse =
s3.getBucketCors(bucketCorsRequest);
           List<CORSRule> corsRules = corsResponse.corsRules();
           for (CORSRule rule : corsRules) {
               System.out.println("allowOrigins: " + rule.allowedOrigins());
               System.out.println("AllowedMethod: " + rule.allowedMethods());
           }

       } catch (S3Exception e) {

           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   public static void setCorsInformation(S3Client s3, String bucketName, String
accountId) {
       List<String> allowMethods = new ArrayList<>();
```

```
        allowMethods.add("PUT");
        allowMethods.add("POST");
        allowMethods.add("DELETE");

        List<String> allowOrigins = new ArrayList<>();
        allowOrigins.add("http://example.com");
        try {
            // Define CORS rules.
            CORSRule corsRule = CORSRule.builder()
                    .allowedMethods(allowMethods)
                    .allowedOrigins(allowOrigins)
                    .build();

            List<CORSRule> corsRules = new ArrayList<>();
            corsRules.add(corsRule);
            CORSConfiguration configuration = CORSConfiguration.builder()
                    .corsRules(corsRules)
                    .build();

            PutBucketCorsRequest putBucketCorsRequest =
 PutBucketCorsRequest.builder()
                    .bucket(bucketName)
                    .corsConfiguration(configuration)
                    .expectedBucketOwner(accountId)
                    .build();

            s3.putBucketCors(putBucketCorsRequest);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat PutBucketCorsdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Tambahkan aturan CORS.

```
import { PutBucketCorsCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

// By default, Amazon S3 doesn't allow cross-origin requests. Use this command
// to explicitly allow cross-origin requests.
export const main = async () => {
  const command = new PutBucketCorsCommand({
    Bucket: "test-bucket",
    CORSConfiguration: {
      CORSRules: [
        {
          // Allow all headers to be sent to this bucket.
          AllowedHeaders: ["*"],
          // Allow only GET and PUT methods to be sent to this bucket.
          AllowedMethods: ["GET", "PUT"],
          // Allow only requests from the specified origin.
          AllowedOrigins: ["https://www.example.com"],
          // Allow the entity tag (ETag) header to be returned in the response.
 The ETag header
          // The entity tag represents a specific version of the object. The ETag
 reflects
          // changes only to the contents of an object, not its metadata.
          ExposeHeaders: ["ETag"],
          // How long the requesting browser should cache the preflight response.
 After
          // this time, the preflight request will have to be made again.
          MaxAgeSeconds: 3600,
        },
      ],
    },
```

```
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat PutBucketCorsdi Referensi AWS SDK for JavaScript API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def put_cors(self, cors_rules):
        """
        Apply CORS rules to the bucket. CORS rules specify the HTTP actions that
 are
```

```
        allowed from other domains.

        :param cors_rules: The CORS rules to apply.
        """
        try:
            self.bucket.Cors().put(CORSConfiguration={"CORSRules": cors_rules})
            logger.info(
                "Put CORS rules %s for bucket '%s'.", cors_rules,
 self.bucket.name
            )
        except ClientError:
            logger.exception("Couldn't put CORS rules for bucket %s.",
 self.bucket.name)
            raise
```

- Untuk detail API, lihat PutBucketCorsdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket CORS configuration.
class BucketCorsWrapper
  attr_reader :bucket_cors

  # @param bucket_cors [Aws::S3::BucketCors] A bucket CORS object configured with
 an existing bucket.
  def initialize(bucket_cors)
    @bucket_cors = bucket_cors
  end

  # Sets CORS rules on a bucket.
```

```
  #
  # @param allowed_methods [Array<String>] The types of HTTP requests to allow.
  # @param allowed_origins [Array<String>] The origins to allow.
  # @returns [Boolean] True if the CORS rules were set; otherwise, false.
  def set_cors(allowed_methods, allowed_origins)
    @bucket_cors.put(
      cors_configuration: {
        cors_rules: [
          {
            allowed_methods: allowed_methods,
            allowed_origins: allowed_origins,
            allowed_headers: %w[*],
            max_age_seconds: 3600
          }
        ]
      }
    )
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't set CORS rules for #{@bucket_cors.bucket.name}. Here's why:
 #{e.message}"
    false
  end

end
```

- Untuk detail API, lihat PutBucketCorsdi Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutBucketEncryption`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketEncryption`.

CLI

AWS CLI

Untuk mengonfigurasi enkripsi sisi server untuk bucket

`put-bucket-encryption`Contoh berikut menetapkan enkripsi AES256 sebagai default untuk bucket yang ditentukan.

```
aws s3api put-bucket-encryption \
    --bucket my-bucket \
    --server-side-encryption-configuration '{"Rules":
 [{"ApplyServerSideEncryptionByDefault": {"SSEAlgorithm": "AES256"}}]}'
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat PutBucketEncryptiondi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengaktifkan enkripsi sisi server AES256 default dengan Amazon S3 Managed Keys (SSE-S3) pada bucket yang diberikan.

```
$Encryptionconfig = @{ServerSideEncryptionByDefault =
 @{ServerSideEncryptionAlgorithm = "AES256"}}
Set-S3BucketEncryption -BucketName 's3testbucket' -
ServerSideEncryptionConfiguration_ServerSideEncryptionRule $Encryptionconfig
```

- Untuk detail API, lihat PutBucketEncryptiondi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutBucketLifecycleConfiguration`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketLifecycleConfiguration`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Hapus unggahan multipart yang tidak lengkap
- Bekerja dengan objek berversi

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        /// <summary>
        /// Adds lifecycle configuration information to the S3 bucket named in
        /// the bucketName parameter.
        /// </summary>
        /// <param name="client">The S3 client used to call the
        /// PutLifecycleConfigurationAsync method.</param>
        /// <param name="bucketName">A string representing the S3 bucket to
        /// which configuration information will be added.</param>
        /// <param name="configuration">A LifecycleConfiguration object that
        /// will be applied to the S3 bucket.</param>
        public static async Task AddExampleLifecycleConfigAsync(IAmazonS3 client,
 string bucketName, LifecycleConfiguration configuration)
        {
            var request = new PutLifecycleConfigurationRequest()
            {
                BucketName = bucketName,
                Configuration = configuration,
            };
            var response = await client.PutLifecycleConfigurationAsync(request);
        }
```

- Untuk detail API, lihat PutBucketLifecycleConfigurationdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Perintah berikut menerapkan konfigurasi siklus hidup ke bucket bernama: `my-bucket`

```
aws s3api put-bucket-lifecycle-configuration --bucket my-bucket --lifecycle-
configuration  file://lifecycle.json
```

File `lifecycle.json` adalah dokumen JSON di folder saat ini yang menentukan dua aturan:

```
{
    "Rules": [
        {
            "ID": "Move rotated logs to Glacier",
            "Prefix": "rotated/",
            "Status": "Enabled",
            "Transitions": [
                {
                    "Date": "2015-11-10T00:00:00.000Z",
                    "StorageClass": "GLACIER"
                }
            ]
        },
        {
            "Status": "Enabled",
            "Prefix": "",
            "NoncurrentVersionTransitions": [
                {
                    "NoncurrentDays": 2,
                    "StorageClass": "GLACIER"
                }
            ],
            "ID": "Move old versions to Glacier"
        }
    ]
}
```

Aturan pertama memindahkan file dengan awalan `rotated` ke Glacier pada tanggal yang ditentukan. Aturan kedua memindahkan versi objek lama ke Glacier ketika mereka tidak lagi terkini. Untuk informasi tentang format stempel waktu yang dapat diterima, lihat Menentukan Nilai Parameter dalam Panduan Pengguna CLI AWS .

- Untuk detail API, lihat PutBucketLifecycleConfigurationdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.LifecycleRuleFilter;
import software.amazon.awssdk.services.s3.model.Transition;
import
 software.amazon.awssdk.services.s3.model.GetBucketLifecycleConfigurationRequest;
import
 software.amazon.awssdk.services.s3.model.GetBucketLifecycleConfigurationResponse;
import software.amazon.awssdk.services.s3.model.DeleteBucketLifecycleRequest;
import software.amazon.awssdk.services.s3.model.TransitionStorageClass;
import software.amazon.awssdk.services.s3.model.LifecycleRule;
import software.amazon.awssdk.services.s3.model.ExpirationStatus;
import software.amazon.awssdk.services.s3.model.BucketLifecycleConfiguration;
import
 software.amazon.awssdk.services.s3.model.PutBucketLifecycleConfigurationRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.util.ArrayList;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class LifecycleConfiguration {
        public static void main(String[] args) {
                final String usage = """
```

```
                                Usage:
                                  <bucketName> <accountId>\s

                                Where:
                                  bucketName - The Amazon Simple Storage Service
(Amazon S3) bucket to upload an object into.
                                  accountId - The id of the account that owns the
Amazon S3 bucket.
                                """;

               if (args.length != 2) {
                       System.out.println(usage);
                       System.exit(1);
               }

               String bucketName = args[0];
               String accountId = args[1];
               Region region = Region.US_EAST_1;
               S3Client s3 = S3Client.builder()
                               .region(region)
                               .build();

               setLifecycleConfig(s3, bucketName, accountId);
               getLifecycleConfig(s3, bucketName, accountId);
               deleteLifecycleConfig(s3, bucketName, accountId);
               System.out.println("You have successfully created, updated, and
deleted a Lifecycle configuration");
               s3.close();
       }

       public static void setLifecycleConfig(S3Client s3, String bucketName,
String accountId) {
               try {
                       // Create a rule to archive objects with the
"glacierobjects/" prefix to Amazon
                       // S3 Glacier.
                       LifecycleRuleFilter ruleFilter =
LifecycleRuleFilter.builder()
                                       .prefix("glacierobjects/")
                                       .build();

                       Transition transition = Transition.builder()
```

```
.storageClass(TransitionStorageClass.GLACIER)
                                        .days(0)
                                        .build();

                        LifecycleRule rule1 = LifecycleRule.builder()
                                        .id("Archive immediately rule")
                                        .filter(ruleFilter)
                                        .transitions(transition)
                                        .status(ExpirationStatus.ENABLED)
                                        .build();

                        // Create a second rule.
                        Transition transition2 = Transition.builder()

.storageClass(TransitionStorageClass.GLACIER)
                                        .days(0)
                                        .build();

                        List<Transition> transitionList = new ArrayList<>();
                        transitionList.add(transition2);

                        LifecycleRuleFilter ruleFilter2 =
LifecycleRuleFilter.builder()
                                        .prefix("glacierobjects/")
                                        .build();

                        LifecycleRule rule2 = LifecycleRule.builder()
                                        .id("Archive and then delete rule")
                                        .filter(ruleFilter2)
                                        .transitions(transitionList)
                                        .status(ExpirationStatus.ENABLED)
                                        .build();

                        // Add the LifecycleRule objects to an ArrayList.
                        ArrayList<LifecycleRule> ruleList = new ArrayList<>();
                        ruleList.add(rule1);
                        ruleList.add(rule2);

                        BucketLifecycleConfiguration lifecycleConfiguration =
BucketLifecycleConfiguration.builder()
                                        .rules(ruleList)
                                        .build();
```

```
                        PutBucketLifecycleConfigurationRequest
putBucketLifecycleConfigurationRequest = PutBucketLifecycleConfigurationRequest
                                        .builder()
                                        .bucket(bucketName)

.lifecycleConfiguration(lifecycleConfiguration)
                                        .expectedBucketOwner(accountId)
                                        .build();


s3.putBucketLifecycleConfiguration(putBucketLifecycleConfigurationRequest);

               } catch (S3Exception e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
               }
       }

       // Retrieve the configuration and add a new rule.
       public static void getLifecycleConfig(S3Client s3, String bucketName,
String accountId) {
               try {
                        GetBucketLifecycleConfigurationRequest
getBucketLifecycleConfigurationRequest = GetBucketLifecycleConfigurationRequest
                                        .builder()
                                        .bucket(bucketName)
                                        .expectedBucketOwner(accountId)
                                        .build();

                        GetBucketLifecycleConfigurationResponse response = s3

.getBucketLifecycleConfiguration(getBucketLifecycleConfigurationRequest);
                        List<LifecycleRule> newList = new ArrayList<>();
                        List<LifecycleRule> rules = response.rules();
                        for (LifecycleRule rule : rules) {
                                newList.add(rule);
                        }

                        // Add a new rule with both a prefix predicate and a tag
predicate.
                        LifecycleRuleFilter ruleFilter =
LifecycleRuleFilter.builder()
                                        .prefix("YearlyDocuments/")
                                        .build();
```

```
                        Transition transition = Transition.builder()

.storageClass(TransitionStorageClass.GLACIER)
                                        .days(3650)
                                        .build();

                        LifecycleRule rule1 = LifecycleRule.builder()
                                        .id("NewRule")
                                        .filter(ruleFilter)
                                        .transitions(transition)
                                        .status(ExpirationStatus.ENABLED)
                                        .build();

                        // Add the new rule to the list.
                        newList.add(rule1);
                        BucketLifecycleConfiguration lifecycleConfiguration =
BucketLifecycleConfiguration.builder()
                                        .rules(newList)
                                        .build();

                        PutBucketLifecycleConfigurationRequest
putBucketLifecycleConfigurationRequest = PutBucketLifecycleConfigurationRequest
                                        .builder()
                                        .bucket(bucketName)

.lifecycleConfiguration(lifecycleConfiguration)
                                        .expectedBucketOwner(accountId)
                                        .build();


s3.putBucketLifecycleConfiguration(putBucketLifecycleConfigurationRequest);

                } catch (S3Exception e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
                }
        }

        // Delete the configuration from the Amazon S3 bucket.
        public static void deleteLifecycleConfig(S3Client s3, String bucketName,
String accountId) {
                try {
```

```
                        DeleteBucketLifecycleRequest deleteBucketLifecycleRequest
 = DeleteBucketLifecycleRequest
                                        .builder()
                                        .bucket(bucketName)
                                        .expectedBucketOwner(accountId)
                                        .build();

                        s3.deleteBucketLifecycle(deleteBucketLifecycleRequest);

                } catch (S3Exception e) {
                        System.err.println(e.awsErrorDetails().errorMessage());
                        System.exit(1);
                }
        }
}
```

- Untuk detail API, lihat PutBucketLifecycleConfigurationdi Referensi AWS SDK for Java 2.x API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name
```

```
    def put_lifecycle_configuration(self, lifecycle_rules):
        """
        Apply a lifecycle configuration to the bucket. The lifecycle
configuration can
        be used to archive or delete the objects in the bucket according to
specified
        parameters, such as a number of days.

        :param lifecycle_rules: The lifecycle rules to apply.
        """
        try:
            self.bucket.LifecycleConfiguration().put(
                LifecycleConfiguration={"Rules": lifecycle_rules}
            )
            logger.info(
                "Put lifecycle rules %s for bucket '%s'.",
                lifecycle_rules,
                self.bucket.name,
            )
        except ClientError:
            logger.exception(
                "Couldn't put lifecycle rules for bucket '%s'.", self.bucket.name
            )
            raise
```

- Untuk detail API, lihat PutBucketLifecycleConfigurationdi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutBucketLogging`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketLogging`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
using System;
using System.IO;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;
using Microsoft.Extensions.Configuration;

/// <summary>
/// This example shows how to enable logging on an Amazon Simple Storage
/// Service (Amazon S3) bucket. You need to have two Amazon S3 buckets for
/// this example. The first is the bucket for which you wish to enable
/// logging, and the second is the location where you want to store the
/// logs.
/// </summary>
public class ServerAccessLogging
{
    private static IConfiguration _configuration = null!;

    public static async Task Main()
    {
        LoadConfig();

        string bucketName = _configuration["BucketName"];
        string logBucketName = _configuration["LogBucketName"];
        string logObjectKeyPrefix = _configuration["LogObjectKeyPrefix"];
        string accountId = _configuration["AccountId"];

        // If the AWS Region defined for your default user is different
        // from the Region where your Amazon S3 bucket is located,
        // pass the Region name to the Amazon S3 client object's constructor.
        // For example: RegionEndpoint.USWest2 or RegionEndpoint.USEast2.
        IAmazonS3 client = new AmazonS3Client();
```

```
            try
            {
                // Update bucket policy for target bucket to allow delivery of
logs to it.
                await SetBucketPolicyToAllowLogDelivery(
                    client,
                    bucketName,
                    logBucketName,
                    logObjectKeyPrefix,
                    accountId);

                // Enable logging on the source bucket.
                await EnableLoggingAsync(
                    client,
                    bucketName,
                    logBucketName,
                    logObjectKeyPrefix);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine($"Error: {e.Message}");
            }
        }

        /// <summary>
        /// This method grants appropriate permissions for logging to the
        /// Amazon S3 bucket where the logs will be stored.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client which will be
used
        /// to apply the bucket policy.</param>
        /// <param name="sourceBucketName">The name of the source bucket.</param>
        /// <param name="logBucketName">The name of the bucket where logging
        /// information will be stored.</param>
        /// <param name="logPrefix">The logging prefix where the logs should be
delivered.</param>
        /// <param name="accountId">The account id of the account where the
source bucket exists.</param>
        /// <returns>Async task.</returns>
        public static async Task SetBucketPolicyToAllowLogDelivery(
            IAmazonS3 client,
            string sourceBucketName,
            string logBucketName,
```

```
            string logPrefix,
            string accountId)
        {
            var resourceArn = @"""arn:aws:s3:::" + logBucketName + "/" +
logPrefix + @"*""";

            var newPolicy = @"{
                                ""Statement"":[{
                                ""Sid"": ""S3ServerAccessLogsPolicy"",
                                ""Effect"": ""Allow"",
                                ""Principal"": { ""Service"":
""logging.s3.amazonaws.com"" },
                                ""Action"": [""s3:PutObject""],
                                ""Resource"": [" + resourceArn + @"],
                                ""Condition"": {
                                ""ArnLike"": { ""aws:SourceArn"":
""arn:aws:s3:::" + sourceBucketName + @""" },
                                ""StringEquals"": { ""aws:SourceAccount"": """ +
accountId + @""" }
                                        }
                                    }]
                                }";
            Console.WriteLine($"The policy to apply to bucket {logBucketName} to
enable logging:");
            Console.WriteLine(newPolicy);

            PutBucketPolicyRequest putRequest = new PutBucketPolicyRequest
            {
                BucketName = logBucketName,
                Policy = newPolicy,
            };
            await client.PutBucketPolicyAsync(putRequest);
            Console.WriteLine("Policy applied.");
        }

        /// <summary>
        /// This method enables logging for an Amazon S3 bucket. Logs will be
stored
        /// in the bucket you selected for logging. Selected prefix
        /// will be prepended to each log object.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client which will be
used
```

```
        /// to configure and apply logging to the selected Amazon S3 bucket.</
param>
        /// <param name="bucketName">The name of the Amazon S3 bucket for which
 you
        /// wish to enable logging.</param>
        /// <param name="logBucketName">The name of the Amazon S3 bucket where
 logging
        /// information will be stored.</param>
        /// <param name="logObjectKeyPrefix">The prefix to prepend to each
        /// object key.</param>
        /// <returns>Async task.</returns>
        public static async Task EnableLoggingAsync(
            IAmazonS3 client,
            string bucketName,
            string logBucketName,
            string logObjectKeyPrefix)
        {
            Console.WriteLine($"Enabling logging for bucket {bucketName}.");
            var loggingConfig = new S3BucketLoggingConfig
            {
                TargetBucketName = logBucketName,
                TargetPrefix = logObjectKeyPrefix,
            };

            var putBucketLoggingRequest = new PutBucketLoggingRequest
            {
                BucketName = bucketName,
                LoggingConfig = loggingConfig,
            };
            await client.PutBucketLoggingAsync(putBucketLoggingRequest);
            Console.WriteLine($"Logging enabled.");
        }

        /// <summary>
        /// Loads configuration from settings files.
        /// </summary>
        public static void LoadConfig()
        {
            _configuration = new ConfigurationBuilder()
                .SetBasePath(Directory.GetCurrentDirectory())
                .AddJsonFile("settings.json") // Load settings from .json file.
                .AddJsonFile("settings.local.json", true) // Optionally, load
 local settings.
                .Build();
```

```
        }
    }
```

- Untuk detail API, lihat PutBucketLoggingdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Contoh 1: Untuk mengatur pencatatan kebijakan bucket

`put-bucket-logging`Contoh berikut menetapkan kebijakan logging untuk MyBucket. Pertama, berikan izin utama layanan logging dalam kebijakan bucket Anda menggunakan `put-bucket-policy` perintah.

```
aws s3api put-bucket-policy \
    --bucket MyBucket \
    --policy file://policy.json
```

Isi dari `policy.json`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ServerAccessLogsPolicy",
            "Effect": "Allow",
            "Principal": {"Service": "logging.s3.amazonaws.com"},
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::MyBucket/Logs/*",
            "Condition": {
                "ArnLike": {"aws:SourceARN": "arn:aws:s3:::SOURCE-BUCKET-NAME"},
                "StringEquals": {"aws:SourceAccount": "SOURCE-AWS-ACCOUNT-ID"}
            }
        }
    ]
}
```

Untuk menerapkan kebijakan pencatatan, gunakan`put-bucket-logging`.

```
aws s3api put-bucket-logging \
    --bucket MyBucket \
    --bucket-logging-status file://logging.json
```

Isi dari `logging.json`:

```
{
     "LoggingEnabled": {
         "TargetBucket": "MyBucket",
         "TargetPrefix": "Logs/"
     }
 }
```

`put-bucket-policy`Perintah diperlukan untuk memberikan `s3:PutObject` izin kepada prinsipal layanan logging.

Untuk informasi selengkapnya, lihat Logging Akses Server Amazon S3 di Panduan Pengguna Amazon S3.

Contoh 2: Untuk menetapkan kebijakan bucket untuk akses log hanya ke satu pengguna

`put-bucket-logging`Contoh berikut menetapkan kebijakan logging untuk MyBucket. AWS Pengguna bob@example.com akan memiliki kontrol penuh atas file log, dan tidak ada orang lain yang memiliki akses. Pertama, berikan izin S3 dengan`put-bucket-acl`.

```
aws s3api put-bucket-acl \
    --bucket MyBucket \
    --grant-write URI=http://acs.amazonaws.com/groups/s3/LogDelivery \
    --grant-read-acp URI=http://acs.amazonaws.com/groups/s3/LogDelivery
```

Kemudian terapkan kebijakan pencatatan menggunakan`put-bucket-logging`.

```
aws s3api put-bucket-logging \
    --bucket MyBucket \
    --bucket-logging-status file://logging.json
```

Isi dari `logging.json`:

```
{
    "LoggingEnabled": {
```

```
        "TargetBucket": "MyBucket",
        "TargetPrefix": "MyBucketLogs/",
        "TargetGrants": [
            {
                "Grantee": {
                    "Type": "AmazonCustomerByEmail",
                    "EmailAddress": "bob@example.com"
                },
                "Permission": "FULL_CONTROL"
            }
        ]
    }
}
```

`put-bucket-acl`perintah diperlukan untuk memberikan sistem pengiriman log S3 izin yang diperlukan (izin tulis dan baca-acp).

Untuk informasi selengkapnya, lihat Logging Akses Server Amazon S3 di Panduan Pengembang Amazon S3.

- Untuk detail API, lihat PutBucketLoggingdi Referensi AWS CLI Perintah.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutBucketNotification`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketNotification`.

CLI

AWS CLI

Menerapkan konfigurasi notifikasi ke bucket bernama`my-bucket`:

```
aws s3api put-bucket-notification --bucket my-bucket --notification-configuration
 file://notification.json
```

File `notification.json` ini adalah dokumen JSON di folder saat ini yang menentukan topik SNS dan jenis acara untuk dipantau:

```
{
  "TopicConfiguration": {
    "Event": "s3:ObjectCreated:*",
    "Topic": "arn:aws:sns:us-west-2:123456789012:s3-notification-topic"
  }
}
```

Topik SNS harus memiliki kebijakan IAM yang melekat padanya yang memungkinkan Amazon S3 untuk mempublikasikannya:

```
{
 "Version": "2008-10-17",
 "Id": "example-ID",
 "Statement": [
  {
   "Sid": "example-statement-ID",
   "Effect": "Allow",
   "Principal": {
     "Service": "s3.amazonaws.com"
   },
   "Action": [
    "SNS:Publish"
   ],
   "Resource": "arn:aws:sns:us-west-2:123456789012:my-bucket",
   "Condition": {
      "ArnLike": {
      "aws:SourceArn": "arn:aws:s3:*:*:my-bucket"
    }
   }
  }
 ]
}
```

- Untuk detail API, lihat PutBucketNotificationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini mengonfigurasi konfigurasi topik SNS untuk acara S3 ObjectRemovedDelete dan mengaktifkan notifikasi untuk bucket s3 yang diberikan

```
$topic =  [Amazon.S3.Model.TopicConfiguration] @{
  Id = "delete-event"
  Topic = "arn:aws:sns:eu-west-1:123456789012:topic-1"
  Event = [Amazon.S3.EventType]::ObjectRemovedDelete
}

Write-S3BucketNotification -BucketName kt-tools -TopicConfiguration $topic
```

Contoh 2: Contoh ini memungkinkan pemberitahuan ObjectCreatedAll untuk bucket yang diberikan mengirimnya ke fungsi Lambda.

```
$lambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
  Events = "s3:ObjectCreated:*"
  FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:rdplock"
  Id = "ObjectCreated-Lambda"
  Filter = @{
    S3KeyFilter = @{
      FilterRules = @(
        @{Name="Prefix";Value="dada"}
        @{Name="Suffix";Value=".pem"}
      )
    }
  }
}

Write-S3BucketNotification -BucketName ssm-editor -LambdaFunctionConfiguration
 $lambdaConfig
```

Contoh 3: Contoh ini membuat 2 konfigurasi Lambda yang berbeda berdasarkan akhiran kunci yang berbeda dan dikonfigurasi keduanya dalam satu perintah.

```
#Lambda Config 1

$firstLambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
  Events = "s3:ObjectCreated:*"
  FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:verifynet"
  Id = "ObjectCreated-dada-ps1"
  Filter = @{
    S3KeyFilter = @{
      FilterRules = @(
        @{Name="Prefix";Value="dada"}
        @{Name="Suffix";Value=".ps1"}
```

```
      )
    }
  }
}

#Lambda Config 2

$secondlambdaConfig = [Amazon.S3.Model.LambdaFunctionConfiguration] @{
  Events = [Amazon.S3.EventType]::ObjectCreatedAll
  FunctionArn = "arn:aws:lambda:eu-west-1:123456789012:function:verifyssm"
  Id = "ObjectCreated-dada-json"
  Filter = @{
    S3KeyFilter = @{
      FilterRules = @(
        @{Name="Prefix";Value="dada"}
        @{Name="Suffix";Value=".json"}
      )
    }
  }
}

Write-S3BucketNotification -BucketName ssm-editor -LambdaFunctionConfiguration
 $firstLambdaConfig,$secondlambdaConfig
```

- Untuk detail API, lihat PutBucketNotificationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutBucketNotificationConfiguration`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketNotificationConfiguration`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    using System;
    using System.Collections.Generic;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example shows how to enable notifications for an Amazon Simple
    /// Storage Service (Amazon S3) bucket.
    /// </summary>
    public class EnableNotifications
    {
        public static async Task Main()
        {
            const string bucketName = "doc-example-bucket1";
            const string snsTopic = "arn:aws:sns:us-east-2:0123456789ab:bucket-
notify";
            const string sqsQueue = "arn:aws:sqs:us-
east-2:0123456789ab:Example_Queue";

            IAmazonS3 client = new AmazonS3Client(Amazon.RegionEndpoint.USEast2);
            await EnableNotificationAsync(client, bucketName, snsTopic,
 sqsQueue);
        }

        /// <summary>
        /// This method makes the call to the PutBucketNotificationAsync method.
        /// </summary>
        /// <param name="client">An initialized Amazon S3 client used to call
        /// the PutBucketNotificationAsync method.</param>
        /// <param name="bucketName">The name of the bucket for which
        /// notifications will be turned on.</param>
```

```
        /// <param name="snsTopic">The ARN for the Amazon Simple Notification
        /// Service (Amazon SNS) topic associated with the S3 bucket.</param>
        /// <param name="sqsQueue">The ARN of the Amazon Simple Queue Service
        /// (Amazon SQS) queue to which notifications will be pushed.</param>
        public static async Task EnableNotificationAsync(
            IAmazonS3 client,
            string bucketName,
            string snsTopic,
            string sqsQueue)
        {
            try
            {
                // The bucket for which we are setting up notifications.
                var request = new PutBucketNotificationRequest()
                {
                    BucketName = bucketName,
                };

                // Defines the topic to use when sending a notification.
                var topicConfig = new TopicConfiguration()
                {
                    Events = new List<EventType> { EventType.ObjectCreatedCopy },
                    Topic = snsTopic,
                };
                request.TopicConfigurations = new List<TopicConfiguration>
                {
                    topicConfig,
                };
                request.QueueConfigurations = new List<QueueConfiguration>
                {
                    new QueueConfiguration()
                    {
                        Events = new List<EventType>
{ EventType.ObjectCreatedPut },
                        Queue = sqsQueue,
                    },
                };

                // Now apply the notification settings to the bucket.
                PutBucketNotificationResponse response = await
client.PutBucketNotificationAsync(request);
            }
            catch (AmazonS3Exception ex)
            {
```

```
                Console.WriteLine($"Error: {ex.Message}");
            }
        }
    }
```

- Untuk detail API, lihat PutBucketNotificationConfigurationdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk mengaktifkan notifikasi yang ditentukan ke bucket

`put-bucket-notification-configuration`Contoh berikut menerapkan konfigurasi notifikasi ke bucket bernama`my-bucket`. File tersebut `notification.json` adalah dokumen JSON di folder saat ini yang menentukan topik SNS dan jenis acara untuk dipantau.

```
aws s3api put-bucket-notification-configuration \
    --bucket my-bucket \
    --notification-configuration file://notification.json
```

Isi dari `notification.json`:

```
{
    "TopicConfigurations": [
        {
            "TopicArn": "arn:aws:sns:us-west-2:123456789012:s3-notification-
topic",
            "Events": [
                "s3:ObjectCreated:*"
            ]
        }
    ]
}
```

Topik SNS harus memiliki kebijakan IAM yang melekat padanya yang memungkinkan Amazon S3 untuk mempublikasikannya.

```
{
    "Version": "2008-10-17",
    "Id": "example-ID",
    "Statement": [
        {
            "Sid": "example-statement-ID",
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "SNS:Publish"
            ],
            "Resource": "arn:aws:sns:us-west-2:123456789012::s3-notification-
topic",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:s3:*:*:my-bucket"
                }
            }
        }
    ]
}
```

- Untuk detail API, lihat PutBucketNotificationConfigurationdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.Event;
import software.amazon.awssdk.services.s3.model.NotificationConfiguration;
```

```
import
 software.amazon.awssdk.services.s3.model.PutBucketNotificationConfigurationRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.TopicConfiguration;
import java.util.ArrayList;
import java.util.List;

public class SetBucketEventBridgeNotification {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName>\s

                Where:
                    bucketName - The Amazon S3 bucket.\s
                    topicArn - The Simple Notification Service topic ARN.\s
                    id - An id value used for the topic configuration. This value
 is displayed in the AWS Management Console.\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String topicArn = args[1];
        String id = args[2];
        Region region = Region.US_EAST_1;
        S3Client s3Client = S3Client.builder()
                .region(region)
                .build();

        setBucketNotification(s3Client, bucketName, topicArn, id);
        s3Client.close();
    }

    public static void setBucketNotification(S3Client s3Client, String
 bucketName, String topicArn, String id) {
        try {
            List<Event> events = new ArrayList<>();
            events.add(Event.S3_OBJECT_CREATED_PUT);
```

```
            TopicConfiguration config = TopicConfiguration.builder()
                    .topicArn(topicArn)
                    .events(events)
                    .id(id)
                    .build();

            List<TopicConfiguration> topics = new ArrayList<>();
            topics.add(config);

            NotificationConfiguration configuration =
 NotificationConfiguration.builder()
                    .topicConfigurations(topics)
                    .build();

            PutBucketNotificationConfigurationRequest configurationRequest =
 PutBucketNotificationConfigurationRequest
                    .builder()
                    .bucket(bucketName)
                    .notificationConfiguration(configuration)
                    .skipDestinationValidation(true)
                    .build();

            // Set the bucket notification configuration.
            s3Client.putBucketNotificationConfiguration(configurationRequest);
            System.out.println("Added bucket " + bucketName + " with EventBridge
 events enabled.");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat PutBucketNotificationConfigurationdi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **PutBucketPolicy** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketPolicy`.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::putBucketPolicy(const Aws::String &bucketName,
                                 const Aws::String &policyBody,
                                 const Aws::S3::S3ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client s3Client(clientConfig);

    std::shared_ptr<Aws::StringStream> request_body =
            Aws::MakeShared<Aws::StringStream>("");
    *request_body << policyBody;

    Aws::S3::Model::PutBucketPolicyRequest request;
    request.SetBucket(bucketName);
    request.SetBody(request_body);

    Aws::S3::Model::PutBucketPolicyOutcome outcome =
            s3Client.PutBucketPolicy(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: putBucketPolicy: "
                  << outcome.GetError().GetMessage() << std::endl;
    } else {
        std::cout << "Set the following policy body for the bucket '" <<
                  bucketName << "':" << std::endl << std::endl;
        std::cout << policyBody << std::endl;
    }

    return outcome.IsSuccess();
}
```

```
//! Build a policy JSON string.
/*!
  \param userArn: Aws user Amazon Resource Name (ARN).
      For more information, see https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference_identifiers.html#identifiers-arns.
  \param bucketName: Name of a bucket.
  \return String: Policy as JSON string.
*/

Aws::String getPolicyString(const Aws::String &userArn,
                            const Aws::String &bucketName) {
    return
            "{\n"
            "   \"Version\":\"2012-10-17\",\n"
            "   \"Statement\":[\n"
            "       {\n"
            "           \"Sid\": \"1\",\n"
            "           \"Effect\": \"Allow\",\n"
            "           \"Principal\": {\n"
            "               \"AWS\": \""
            + userArn +
            "\"\n""           },\n"
            "           \"Action\": [ \"s3:getObject\" ],\n"
            "           \"Resource\": [ \"arn:aws:s3:::"
            + bucketName +
            "/*\" ]\n"
            "       }\n"
            "   ]\n"
            "}";
}
```

- Untuk detail API, lihat PutBucketPolicydi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Contoh ini memungkinkan semua pengguna untuk mengambil objek apa pun MyBucketkecuali yang ada di. MySecretFolder Ini juga memberikan `put` dan `delete` izin kepada pengguna root AWS akun`1234-5678-9012`:

```
aws s3api put-bucket-policy --bucket MyBucket --policy file://policy.json

policy.json:
{
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": "*",
         "Action": "s3:GetObject",
         "Resource": "arn:aws:s3:::MyBucket/*"
      },
      {
         "Effect": "Deny",
         "Principal": "*",
         "Action": "s3:GetObject",
         "Resource": "arn:aws:s3:::MyBucket/MySecretFolder/*"
      },
      {
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::123456789012:root"
         },
         "Action": [
            "s3:DeleteObject",
            "s3:PutObject"
         ],
         "Resource": "arn:aws:s3:::MyBucket/*"
      }
   ]
}
```

- Untuk detail API, lihat PutBucketPolicydi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutBucketPolicyRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.List;
import com.fasterxml.jackson.core.JsonParser;
import com.fasterxml.jackson.databind.ObjectMapper;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class SetBucketPolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <polFile>

                Where:
                    bucketName - The Amazon S3 bucket to set the policy on.
                    polFile - A JSON file containing the policy (see the Amazon
 S3 Readme for an example).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String polFile = args[1];
        String policyText = getBucketPolicyFromFile(polFile);
        Region region = Region.US_EAST_1;
```

```
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        setPolicy(s3, bucketName, policyText);
        s3.close();
    }

    public static void setPolicy(S3Client s3, String bucketName, String
policyText) {
        System.out.println("Setting policy:");
        System.out.println("----");
        System.out.println(policyText);
        System.out.println("----");
        System.out.format("On Amazon S3 bucket: \"%s\"\n", bucketName);

        try {
            PutBucketPolicyRequest policyReq = PutBucketPolicyRequest.builder()
                    .bucket(bucketName)
                    .policy(policyText)
                    .build();

            s3.putBucketPolicy(policyReq);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

        System.out.println("Done!");
    }

    // Loads a JSON-formatted policy from a file
    public static String getBucketPolicyFromFile(String policyFile) {

        StringBuilder fileText = new StringBuilder();
        try {
            List<String> lines = Files.readAllLines(Paths.get(policyFile),
StandardCharsets.UTF_8);
            for (String line : lines) {
                fileText.append(line);
            }

        } catch (IOException e) {
```

```
            System.out.format("Problem reading file: \"%s\"", policyFile);
            System.out.println(e.getMessage());
        }

        try {
            final JsonParser parser = new
 ObjectMapper().getFactory().createParser(fileText.toString());
            while (parser.nextToken() != null) {
            }

        } catch (IOException jpe) {
            jpe.printStackTrace();
        }
        return fileText.toString();
    }
}
```

- Untuk detail API, lihat PutBucketPolicydi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Tambahkan kebijakan.

```
import { PutBucketPolicyCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new PutBucketPolicyCommand({
    Policy: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
```

```
          Sid: "AllowGetObject",
          // Allow this particular user to call GetObject on any object in this
 bucket.
          Effect: "Allow",
          Principal: {
            AWS: "arn:aws:iam::ACCOUNT-ID:user/USERNAME",
          },
          Action: "s3:GetObject",
          Resource: "arn:aws:s3:::BUCKET-NAME/*",
        },
      ],
    }),
    // Apply the preceding policy to this bucket.
    Bucket: "BUCKET-NAME",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat PutBucketPolicydi Referensi AWS SDK for JavaScript API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""
```

```
    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def put_policy(self, policy):
        """
        Apply a security policy to the bucket. Policies control users' ability
        to perform specific actions, such as listing the objects in the bucket.

        :param policy: The policy to apply to the bucket.
        """
        try:
            self.bucket.Policy().put(Policy=json.dumps(policy))
            logger.info("Put policy %s for bucket '%s'.", policy,
self.bucket.name)
        except ClientError:
            logger.exception("Couldn't apply policy to bucket '%s'.",
self.bucket.name)
            raise
```

- Untuk detail API, lihat PutBucketPolicydi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# Wraps an Amazon S3 bucket policy.
class BucketPolicyWrapper
```

```
  attr_reader :bucket_policy

  # @param bucket_policy [Aws::S3::BucketPolicy] A bucket policy object
 configured with an existing bucket.
  def initialize(bucket_policy)
    @bucket_policy = bucket_policy
  end

  # Sets a policy on a bucket.
  #
  def set_policy(policy)
    @bucket_policy.put(policy: policy)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't set the policy for #{@bucket_policy.bucket.name}. Here's why:
 #{e.message}"
    false
  end

end
```

- Untuk detail API, lihat PutBucketPolicydi Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutBucketReplication`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketReplication`.

CLI

AWS CLI

Untuk mengonfigurasi replikasi untuk bucket S3

`put-bucket-replication`Contoh berikut menerapkan konfigurasi replikasi ke bucket S3 yang ditentukan.

```
aws s3api put-bucket-replication \
```

```
    --bucket AWSDOC-EXAMPLE-BUCKET1 \
    --replication-configuration file://replication.json
```

Isi dari `replication.json`:

```
{
    "Role": "arn:aws:iam::123456789012:role/s3-replication-role",
    "Rules": [
        {
            "Status": "Enabled",
            "Priority": 1,
            "DeleteMarkerReplication": { "Status": "Disabled" },
            "Filter" : { "Prefix": ""},
            "Destination": {
                "Bucket": "arn:aws:s3:::AWSDOC-EXAMPLE-BUCKET2"
            }
        }
    ]
}
```

Bucket tujuan harus mengaktifkan versi. Peran yang ditentukan harus memiliki izin untuk menulis ke bucket tujuan dan memiliki hubungan kepercayaan yang memungkinkan Amazon S3 untuk mengambil peran tersebut.

Contoh kebijakan izin peran:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetReplicationConfiguration",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::AWSDOC-EXAMPLE-BUCKET1"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "s3:GetObjectVersion",
                "s3:GetObjectVersionAcl",
                "s3:GetObjectVersionTagging"
            ],
            "Resource": [
                "arn:aws:s3:::AWSDOC-EXAMPLE-BUCKET1/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ReplicateObject",
                "s3:ReplicateDelete",
                "s3:ReplicateTags"
            ],
            "Resource": "arn:aws:s3:::AWSDOC-EXAMPLE-BUCKET2/*"
        }
    ]
}
```

Contoh kebijakan hubungan kepercayaan:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Perintah ini tidak menghasilkan output.

Untuk informasi selengkapnya, lihat Ini adalah judul topik di Panduan Pengguna Konsol Layanan Penyimpanan Sederhana Amazon.

- Untuk detail API, lihat PutBucketReplicationdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menetapkan konfigurasi replikasi dengan satu aturan yang memungkinkan replikasi ke bucket 'exampletargetbucket' setiap objek baru yang dibuat dengan awalan nama kunci "" di bucket 'examplebucket'. TaxDocs

```
$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Enabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::exampletargetbucket" }

$params = @{
    BucketName = "examplebucket"
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1
}

Write-S3BucketReplication @params
```

Contoh 2: Contoh ini menetapkan konfigurasi replikasi dengan beberapa aturan yang memungkinkan replikasi ke bucket 'exampletargetbucket' setiap objek baru yang dibuat dengan awalan nama kunci "" atau "". TaxDocs OtherDocs Awalan kunci tidak boleh tumpang tindih.

```
$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Enabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::exampletargetbucket" }

$rule2 = New-Object Amazon.S3.Model.ReplicationRule
$rule2.ID = "Rule-2"
$rule2.Status = "Enabled"
$rule2.Prefix = "OtherDocs"
$rule2.Destination = @{ BucketArn = "arn:aws:s3:::exampletargetbucket" }

$params = @{
    BucketName = "examplebucket"
```

```
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1,$rule2
}

Write-S3BucketReplication @params
```

Contoh 3: Contoh ini memperbarui konfigurasi replikasi pada bucket yang ditentukan untuk menonaktifkan aturan yang mengontrol replikasi objek dengan awalan nama kunci "TaxDocs" ke bucket 'exampletargetbucket'.

```
$rule1 = New-Object Amazon.S3.Model.ReplicationRule
$rule1.ID = "Rule-1"
$rule1.Status = "Disabled"
$rule1.Prefix = "TaxDocs"
$rule1.Destination = @{ BucketArn = "arn:aws:s3:::exampletargetbucket" }

$params = @{
    BucketName = "examplebucket"
    Configuration_Role = "arn:aws:iam::35667example:role/
CrossRegionReplicationRoleForS3"
    Configuration_Rule = $rule1
}

Write-S3BucketReplication @params
```

- Untuk detail API, lihat PutBucketReplicationdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutBucketRequestPayment`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketRequestPayment`.

CLI

AWS CLI

Contoh 1: Untuk mengaktifkan konfigurasi ``requester pays`` untuk bucket

`put-bucket-request-payment`Contoh berikut memungkinkan `requester pays` untuk bucket yang ditentukan.

```
aws s3api put-bucket-request-payment \
    --bucket my-bucket \
    --request-payment-configuration '{"Payer":"Requester"}'
```

Perintah ini tidak menghasilkan output.

Contoh 2: Untuk menonaktifkan konfigurasi ``requester pays`` untuk bucket

`put-bucket-request-payment`Contoh berikut menonaktifkan `requester pays` untuk bucket yang ditentukan.

```
aws s3api put-bucket-request-payment \
    --bucket my-bucket \
    --request-payment-configuration '{"Payer":"BucketOwner"}'
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat PutBucketRequestPaymentdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Memperbarui konfigurasi pembayaran permintaan untuk bucket bernama 'mybucket' sehingga orang yang meminta unduhan dari bucket akan dikenakan biaya untuk unduhan. Secara default, pemilik bucket membayar unduhan. Untuk mengatur permintaan pembayaran kembali ke default gunakan 'BucketOwner' untuk parameter RequestPaymentConfiguration _Payer.

```
Write-S3BucketRequestPayment -BucketName mybucket -
RequestPaymentConfiguration_Payer Requester
```

- Untuk detail API, lihat PutBucketRequestPaymentdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutBucketTagging`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketTagging`.

CLI

AWS CLI

Perintah berikut menerapkan konfigurasi penandaan ke bucket bernama`my-bucket`:

```
aws s3api put-bucket-tagging --bucket my-bucket --tagging file://tagging.json
```

File `tagging.json` adalah dokumen JSON di folder saat ini yang menentukan tag:

```
{
   "TagSet": [
     {
       "Key": "organization",
       "Value": "marketing"
     }
   ]
}
```

Atau terapkan konfigurasi penandaan `my-bucket` langsung dari baris perintah:

```
aws s3api put-bucket-tagging --bucket my-bucket --tagging
 'TagSet=[{Key=organization,Value=marketing}]'
```

- Untuk detail API, lihat PutBucketTaggingdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini menerapkan dua tag ke bucket bernama**`cloudtrail-test-2018`**: tag dengan kunci Stage dan nilai Test, dan tag dengan kunci Environment dan nilai Alpha. Untuk memverifikasi bahwa tag telah ditambahkan ke bucket, jalankan**`Get-S3BucketTagging`**

-BucketName bucket_name. Hasilnya harus menunjukkan tag yang Anda terapkan ke bucket di perintah pertama. Perhatikan bahwa **Write-S3BucketTagging** menimpa seluruh set tag yang ada di bucket. Untuk menambah atau menghapus tag individual, jalankan cmdlet Resource Groups dan Tagging API, dan. **Add-RGTResourceTag Remove-RGTResourceTag** Atau, gunakan Editor Tag di AWS Management Console untuk mengelola tag bucket S3.

```
Write-S3BucketTagging -BucketName cloudtrail-test-2018 -TagSet @( @{ Key="Stage";
 Value="Test" }, @{ Key="Environment"; Value="Alpha" } )
```

Contoh 2: Perintah ini menyalurkan ember yang diberi nama **cloudtrail-test-2018** ke dalam **Write-S3BucketTagging** cmdlet. Ini berlaku tag Tahap: Produksi dan Departemen: Keuangan ke ember. Perhatikan bahwa **Write-S3BucketTagging** menimpa seluruh set tag yang ada di bucket.

```
Get-S3Bucket -BucketName cloudtrail-test-2018 | Write-S3BucketTagging
 -TagSet @( @{ Key="Stage"; Value="Production" }, @{ Key="Department";
 Value="Finance" } )
```

- Untuk detail API, lihat PutBucketTaggingdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **PutBucketVersioning** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketVersioning`.

CLI

AWS CLI

Perintah berikut memungkinkan pembuatan versi pada bucket bernama: `my-bucket`

```
aws s3api put-bucket-versioning --bucket my-bucket --versioning-configuration
 Status=Enabled
```

Perintah berikut memungkinkan pembuatan versi, dan menggunakan kode mfa

```
aws s3api put-bucket-versioning --bucket my-bucket --versioning-configuration
 Status=Enabled --mfa "SERIAL 123456"
```

- Untuk detail API, lihat PutBucketVersioningdi Referensi AWS CLI Perintah.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah memungkinkan pembuatan versi untuk bucket S3 yang diberikan.

```
Write-S3BucketVersioning -BucketName 's3testbucket' -VersioningConfig_Status
 Enabled
```

- Untuk detail API, lihat PutBucketVersioningdi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutBucketWebsite`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutBucketWebsite`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
                // Put the website configuration.
                PutBucketWebsiteRequest putRequest = new
 PutBucketWebsiteRequest()
                {
                    BucketName = bucketName,
```

```
                    WebsiteConfiguration = new WebsiteConfiguration()
                    {
                        IndexDocumentSuffix = indexDocumentSuffix,
                        ErrorDocument = errorDocument,
                    },
                };
                PutBucketWebsiteResponse response = await
 client.PutBucketWebsiteAsync(putRequest);
```

- Untuk detail API, lihat PutBucketWebsitedi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::putWebsiteConfig(const Aws::String &bucketName,
                                  const Aws::String &indexPage, const Aws::String
 &errorPage,
                                  const Aws::S3::S3ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client client(clientConfig);

    Aws::S3::Model::IndexDocument indexDocument;
    indexDocument.SetSuffix(indexPage);

    Aws::S3::Model::ErrorDocument errorDocument;
    errorDocument.SetKey(errorPage);

    Aws::S3::Model::WebsiteConfiguration websiteConfiguration;
    websiteConfiguration.SetIndexDocument(indexDocument);
    websiteConfiguration.SetErrorDocument(errorDocument);

    Aws::S3::Model::PutBucketWebsiteRequest request;
    request.SetBucket(bucketName);
```

```
    request.SetWebsiteConfiguration(websiteConfiguration);

    Aws::S3::Model::PutBucketWebsiteOutcome outcome =
            client.PutBucketWebsite(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: PutBucketWebsite: "
                  << outcome.GetError().GetMessage() << std::endl;
    } else {
        std::cout << "Success: Set website configuration for bucket '"
                  << bucketName << "'." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat PutBucketWebsitedi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Menerapkan konfigurasi situs web statis ke bucket bernama`my-bucket`:

```
aws s3api put-bucket-website --bucket my-bucket --website-configuration file://
website.json
```

File tersebut `website.json` adalah dokumen JSON di folder saat ini yang menentukan halaman indeks dan kesalahan untuk situs web:

```
{
    "IndexDocument": {
        "Suffix": "index.html"
    },
    "ErrorDocument": {
        "Key": "error.html"
    }
}
```

- Untuk detail API, lihat PutBucketWebsitedi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.IndexDocument;
import software.amazon.awssdk.services.s3.model.PutBucketWebsiteRequest;
import software.amazon.awssdk.services.s3.model.WebsiteConfiguration;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.regions.Region;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class SetWebsiteConfiguration {
    public static void main(String[] args) {
        final String usage = """

                Usage:    <bucketName> [indexdoc]\s

                Where:
                   bucketName   - The Amazon S3 bucket to set the website
 configuration on.\s
                   indexdoc - The index document, ex. 'index.html'
                              If not specified, 'index.html' will be set.
                """;

        if (args.length != 1) {
            System.out.println(usage);
```

```
            System.exit(1);
        }

        String bucketName = args[0];
        String indexDoc = "index.html";
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        setWebsiteConfig(s3, bucketName, indexDoc);
        s3.close();
    }

    public static void setWebsiteConfig(S3Client s3, String bucketName, String
 indexDoc) {
        try {
            WebsiteConfiguration websiteConfig = WebsiteConfiguration.builder()

 .indexDocument(IndexDocument.builder().suffix(indexDoc).build())
                    .build();

            PutBucketWebsiteRequest pubWebsiteReq =
 PutBucketWebsiteRequest.builder()
                    .bucket(bucketName)
                    .websiteConfiguration(websiteConfig)
                    .build();

            s3.putBucketWebsite(pubWebsiteReq);
            System.out.println("The call was successful");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat PutBucketWebsitedi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Atur konfigurasi situs web.

```
import { PutBucketWebsiteCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

// Set up a bucket as a static website.
// The bucket needs to be publicly accessible.
export const main = async () => {
  const command = new PutBucketWebsiteCommand({
    Bucket: "test-bucket",
    WebsiteConfiguration: {
      ErrorDocument: {
        // The object key name to use when a 4XX class error occurs.
        Key: "error.html",
      },
      IndexDocument: {
        // A suffix that is appended to a request that is for a directory.
        Suffix: "index.html",
      },
    },
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat PutBucketWebsitedi Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah memungkinkan hosting situs web untuk bucket yang diberikan dengan dokumen indeks sebagai 'index.html' dan dokumen kesalahan sebagai 'error.html'.

```
Write-S3BucketWebsite -BucketName 's3testbucket' -
WebsiteConfiguration_IndexDocumentSuffix 'index.html' -
WebsiteConfiguration_ErrorDocument 'error.html'
```

- Untuk detail API, lihat PutBucketWebsitedi Referensi AWS Tools for PowerShell Cmdlet.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket website actions.
class BucketWebsiteWrapper
  attr_reader :bucket_website

  # @param bucket_website [Aws::S3::BucketWebsite] A bucket website object
 configured with an existing bucket.
  def initialize(bucket_website)
    @bucket_website = bucket_website
  end

  # Sets a bucket as a static website.
```

```
  #
  # @param index_document [String] The name of the index document for the
 website.
  # @param error_document [String] The name of the error document to show for 4XX
 errors.
  # @return [Boolean] True when the bucket is configured as a website; otherwise,
 false.
  def set_website(index_document, error_document)
    @bucket_website.put(
      website_configuration: {
        index_document: { suffix: index_document },
        error_document: { key: error_document }
      }
    )
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't configure #{@bucket_website.bucket.name} as a website. Here's
 why: #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  index_document = "index.html"
  error_document = "404.html"

  wrapper = BucketWebsiteWrapper.new(Aws::S3::BucketWebsite.new(bucket_name))
  return unless wrapper.set_website(index_document, error_document)

  puts "Successfully configured bucket #{bucket_name} as a static website."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk detail API, lihat PutBucketWebsitedi Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **PutObject** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutObject`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai bucket dan objek
- Lacak unggahan dan unduhan

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
/// <summary>
/// Shows how to upload a file from the local computer to an Amazon S3
/// bucket.
/// </summary>
/// <param name="client">An initialized Amazon S3 client object.</param>
/// <param name="bucketName">The Amazon S3 bucket to which the object
/// will be uploaded.</param>
/// <param name="objectName">The object to upload.</param>
/// <param name="filePath">The path, including file name, of the object
/// on the local computer to upload.</param>
/// <returns>A boolean value indicating the success or failure of the
/// upload procedure.</returns>
public static async Task<bool> UploadFileAsync(
    IAmazonS3 client,
    string bucketName,
    string objectName,
    string filePath)
{
    var request = new PutObjectRequest
    {
```

```
                BucketName = bucketName,
                Key = objectName,
                FilePath = filePath,
            };

            var response = await client.PutObjectAsync(request);
            if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
            {
                Console.WriteLine($"Successfully uploaded {objectName} to
{bucketName}.");
                return true;
            }
            else
            {
                Console.WriteLine($"Could not upload {objectName} to
{bucketName}.");
                return false;
            }
        }
```

Unggah objek dengan enkripsi di sisi server.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example shows how to upload an object to an Amazon Simple Storage
    /// Service (Amazon S3) bucket with server-side encryption enabled.
    /// </summary>
    public class ServerSideEncryption
    {
        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";
            string keyName = "samplefile.txt";

            // If the AWS Region defined for your default user is different
            // from the Region where your Amazon S3 bucket is located,
            // pass the Region name to the Amazon S3 client object's constructor.
```

```
            // For example: RegionEndpoint.USWest2.
            IAmazonS3 client = new AmazonS3Client();

            await WritingAnObjectAsync(client, bucketName, keyName);
        }

        /// <summary>
        /// Upload a sample object include a setting for encryption.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
        /// to upload a file and apply server-side encryption.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket where the
        /// encrypted object will reside.</param>
        /// <param name="keyName">The name for the object that you want to
        /// create in the supplied bucket.</param>
        public static async Task WritingAnObjectAsync(IAmazonS3 client, string
 bucketName, string keyName)
        {
            try
            {
                var putRequest = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                    ContentBody = "sample text",
                    ServerSideEncryptionMethod =
 ServerSideEncryptionMethod.AES256,
                };

                var putResponse = await client.PutObjectAsync(putRequest);

                // Determine the encryption state of an object.
                GetObjectMetadataRequest metadataRequest = new
 GetObjectMetadataRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
                };
                GetObjectMetadataResponse response = await
 client.GetObjectMetadataAsync(metadataRequest);
                ServerSideEncryptionMethod objectEncryption =
 response.ServerSideEncryptionMethod;
```

```
                Console.WriteLine($"Encryption method used: {0}",
objectEncryption.ToString());
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error: '{ex.Message}' when writing an
object");
            }
        }
    }
```

- Untuk detail API, lihat PutObjectdi Referensi AWS SDK for .NET API.

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function copy_file_to_bucket
#
# This function creates a file in the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket to copy the file to.
#       $2 - The path and file name of the local file to copy to the bucket.
```

```
#       $3 - The key (name) to call the copy of the file in the bucket.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function copy_file_to_bucket() {
  local response bucket_name source_file destination_file_name
  bucket_name=$1
  source_file=$2
  destination_file_name=$3

  response=$(aws s3api put-object \
    --bucket "$bucket_name" \
    --body "$source_file" \
    --key "$destination_file_name")

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports put-object operation failed.\n$response"
    return 1
  fi
}
```

- Untuk detail API, lihat PutObjectdi Referensi AWS CLI Perintah.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::putObject(const Aws::String &bucketName,
                           const Aws::String &fileName,
                           const Aws::S3::S3ClientConfiguration &clientConfig) {
    Aws::S3::S3Client s3Client(clientConfig);
```

```
    Aws::S3::Model::PutObjectRequest request;
    request.SetBucket(bucketName);
    //We are using the name of the file as the key for the object in the bucket.
    //However, this is just a string and can be set according to your retrieval
 needs.
    request.SetKey(fileName);

    std::shared_ptr<Aws::IOStream> inputData =
            Aws::MakeShared<Aws::FStream>("SampleAllocationTag",
                                          fileName.c_str(),
                                          std::ios_base::in |
 std::ios_base::binary);

    if (!*inputData) {
        std::cerr << "Error unable to read file " << fileName << std::endl;
        return false;
    }

    request.SetBody(inputData);

    Aws::S3::Model::PutObjectOutcome outcome =
            s3Client.PutObject(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: putObject: " <<
                  outcome.GetError().GetMessage() << std::endl;
    } else {
        std::cout << "Added object '" << fileName << "' to bucket '"
                  << bucketName << "'.";
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat PutObjectdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Contoh berikut menggunakan `put-object` perintah untuk mengunggah objek ke Amazon S3:

```
aws s3api put-object --bucket text-content --key dir-1/my_images.tar.bz2 --body
 my_images.tar.bz2
```

Contoh berikut menunjukkan unggahan file video (File video ditentukan menggunakan sintaks sistem file Windows.):

```
aws s3api put-object --bucket text-content --key dir-1/big-video-file.mp4 --body
 e:\media\videos\f-sharp-3-data-services.mp4
```

Untuk informasi selengkapnya tentang mengunggah objek, lihat Mengunggah Objek di Panduan Pengembang Amazon S3.

- Untuk detail API, lihat PutObjectdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Masukkan objek ke dalam ember dengan menggunakan API tingkat rendah.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// UploadFile reads from a file and puts the data into an object in a bucket.
```

```
func (basics BucketBasics) UploadFile(bucketName string, objectKey string,
 fileName string) error {
 file, err := os.Open(fileName)
 if err != nil {
  log.Printf("Couldn't open file %v to upload. Here's why: %v\n", fileName, err)
 } else {
  defer file.Close()
  _, err = basics.S3Client.PutObject(context.TODO(), &s3.PutObjectInput{
   Bucket: aws.String(bucketName),
   Key:    aws.String(objectKey),
   Body:   file,
  })
  if err != nil {
   log.Printf("Couldn't upload file %v to %v:%v. Here's why: %v\n",
    fileName, bucketName, objectKey, err)
  }
 }
 return err
}
```

Unggah objek ke bucket dengan menggunakan manajer transfer.

```
// S3Actions wraps S3 service actions.
type S3Actions struct {
 S3Client  *s3.Client
 S3Manager *manager.Uploader
}



// UploadObject uses the S3 upload manager to upload an object to a bucket.
func (actor S3Actions) UploadObject(ctx context.Context, bucket string, key
 string, contents string) (string, error) {
 var outKey string
 input := &s3.PutObjectInput{
  Bucket:            aws.String(bucket),
  Key:               aws.String(key),
  Body:              bytes.NewReader([]byte(contents)),
  ChecksumAlgorithm: types.ChecksumAlgorithmSha256,
 }
```

```
 output, err := actor.S3Manager.Upload(ctx, input)
 if err != nil {
  var noBucket *types.NoSuchBucket
  if errors.As(err, &noBucket) {
   log.Printf("Bucket %s does not exist.\n", bucket)
   err = noBucket
  }
 } else {
  err := s3.NewObjectExistsWaiter(actor.S3Client).Wait(ctx, &s3.HeadObjectInput{
   Bucket: aws.String(bucket),
   Key:    aws.String(key),
  }, time.Minute)
  if err != nil {
   log.Printf("Failed attempt to wait for object %s to exist in %s.\n", key,
 bucket)
  } else {
   outKey = *output.Key
  }
 }
 return outKey, err
}
```

- Untuk detail API, lihat PutObjectdi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Unggah file ke bucket menggunakan S3Client.

```
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
```

```
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.io.File;
import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class PutObject {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                  <bucketName> <objectKey> <objectPath>\s

                Where:
                  bucketName - The Amazon S3 bucket to upload an object into.
                  objectKey - The object to upload (for example, book.pdf).
                  objectPath - The path where the file is located (for example,
 C:/AWS/book2.pdf).\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String objectKey = args[1];
        String objectPath = args[2];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        putS3Object(s3, bucketName, objectKey, objectPath);
        s3.close();
```

```
    }

    // This example uses RequestBody.fromFile to avoid loading the whole file
 into
    // memory.
    public static void putS3Object(S3Client s3, String bucketName, String
 objectKey, String objectPath) {
        try {
            Map<String, String> metadata = new HashMap<>();
            metadata.put("x-amz-meta-myVal", "test");
            PutObjectRequest putOb = PutObjectRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .metadata(metadata)
                    .build();

            s3.putObject(putOb, RequestBody.fromFile(new File(objectPath)));
            System.out.println("Successfully placed " + objectKey + " into bucket
 " + bucketName);

        } catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

Gunakan S3 TransferManager untuk mengunggah file ke bucket. Lihat file lengkap dan lakukan pengujian.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedFileUpload;
import software.amazon.awssdk.transfer.s3.model.FileUpload;
import software.amazon.awssdk.transfer.s3.model.UploadFileRequest;
import software.amazon.awssdk.transfer.s3.progress.LoggingTransferListener;
import java.net.URI;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.file.Paths;
import java.util.UUID;
```

```
   public String uploadFile(S3TransferManager transferManager, String
bucketName,
                            String key, URI filePathURI) {
       UploadFileRequest uploadFileRequest = UploadFileRequest.builder()
           .putObjectRequest(b -> b.bucket(bucketName).key(key))
           .source(Paths.get(filePathURI))
           .build();

       FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);

       CompletedFileUpload uploadResult = fileUpload.completionFuture().join();
       return uploadResult.response().eTag();
   }
```

Unggah objek ke bucket dan tetapkan tanda menggunakan S3Client.

```
   public static void putS3ObjectTags(S3Client s3, String bucketName, String
objectKey, String objectPath) {
       try {
           Tag tag1 = Tag.builder()
                   .key("Tag 1")
                   .value("This is tag 1")
                   .build();

           Tag tag2 = Tag.builder()
                   .key("Tag 2")
                   .value("This is tag 2")
                   .build();

           List<Tag> tags = new ArrayList<>();
           tags.add(tag1);
           tags.add(tag2);

           Tagging allTags = Tagging.builder()
                   .tagSet(tags)
                   .build();

           PutObjectRequest putOb = PutObjectRequest.builder()
                   .bucket(bucketName)
                   .key(objectKey)
                   .tagging(allTags)
```

```
                    .build();

            s3.putObject(putOb,
RequestBody.fromBytes(getObjectFile(objectPath)));

        } catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void updateObjectTags(S3Client s3, String bucketName, String
objectKey) {
        try {
            GetObjectTaggingRequest taggingRequest =
GetObjectTaggingRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .build();

            GetObjectTaggingResponse getTaggingRes =
s3.getObjectTagging(taggingRequest);
            List<Tag> obTags = getTaggingRes.tagSet();
            for (Tag sinTag : obTags) {
                System.out.println("The tag key is: " + sinTag.key());
                System.out.println("The tag value is: " + sinTag.value());
            }

            // Replace the object's tags with two new tags.
            Tag tag3 = Tag.builder()
                    .key("Tag 3")
                    .value("This is tag 3")
                    .build();

            Tag tag4 = Tag.builder()
                    .key("Tag 4")
                    .value("This is tag 4")
                    .build();

            List<Tag> tags = new ArrayList<>();
            tags.add(tag3);
            tags.add(tag4);

            Tagging updatedTags = Tagging.builder()
```

```
                    .tagSet(tags)
                    .build();

            PutObjectTaggingRequest taggingRequest1 =
PutObjectTaggingRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .tagging(updatedTags)
                    .build();

            s3.putObjectTagging(taggingRequest1);
            GetObjectTaggingResponse getTaggingRes2 =
s3.getObjectTagging(taggingRequest);
            List<Tag> modTags = getTaggingRes2.tagSet();
            for (Tag sinTag : modTags) {
                System.out.println("The tag key is: " + sinTag.key());
                System.out.println("The tag value is: " + sinTag.value());
            }

        } catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    // Return a byte array.
    private static byte[] getObjectFile(String filePath) {
        FileInputStream fileInputStream = null;
        byte[] bytesArray = null;

        try {
            File file = new File(filePath);
            bytesArray = new byte[(int) file.length()];
            fileInputStream = new FileInputStream(file);
            fileInputStream.read(bytesArray);

        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (fileInputStream != null) {
                try {
                    fileInputStream.close();
                } catch (IOException e) {
                    e.printStackTrace();
```

```
                }
            }
        }

        return bytesArray;
    }
}
```

Unggah objek ke bucket dan tetapkan metadata menggunakan S3Client.

```
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.io.File;
import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class PutObjectMetadata {
    public static void main(String[] args) {
        final String USAGE = """

                Usage:
                  <bucketName> <objectKey> <objectPath>\s

                Where:
                  bucketName - The Amazon S3 bucket to upload an object into.
                  objectKey - The object to upload (for example, book.pdf).
                  objectPath - The path where the file is located (for example,
 C:/AWS/book2.pdf).\s
                """;
```

```
        if (args.length != 3) {
            System.out.println(USAGE);
            System.exit(1);
        }

        String bucketName = args[0];
        String objectKey = args[1];
        String objectPath = args[2];
        System.out.println("Putting object " + objectKey + " into bucket " +
 bucketName);
        System.out.println("  in bucket: " + bucketName);
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        putS3Object(s3, bucketName, objectKey, objectPath);
        s3.close();
    }

    // This example uses RequestBody.fromFile to avoid loading the whole file
 into
    // memory.
    public static void putS3Object(S3Client s3, String bucketName, String
 objectKey, String objectPath) {
        try {
            Map<String, String> metadata = new HashMap<>();
            metadata.put("author", "Mary Doe");
            metadata.put("version", "1.0.0.0");

            PutObjectRequest putOb = PutObjectRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .metadata(metadata)
                    .build();

            s3.putObject(putOb, RequestBody.fromFile(new File(objectPath)));
            System.out.println("Successfully placed " + objectKey + " into bucket
 " + bucketName);

        } catch (S3Exception e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
```

```
    }
}
```

Unggah objek ke bucket dan tetapkan nilai retensi objek menggunakan S3Client.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRetentionRequest;
import software.amazon.awssdk.services.s3.model.ObjectLockRetention;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.time.Instant;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.ZoneOffset;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class PutObjectRetention {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <key> <bucketName>\s

                Where:
                    key - The name of the object (for example, book.pdf).\s
                    bucketName - The Amazon S3 bucket name that contains the
 object (for example, bucket1).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }
```

```
        String key = args[0];
        String bucketName = args[1];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        setRentionPeriod(s3, key, bucketName);
        s3.close();
    }

    public static void setRentionPeriod(S3Client s3, String key, String bucket) {
        try {
            LocalDate localDate = LocalDate.parse("2020-07-17");
            LocalDateTime localDateTime = localDate.atStartOfDay();
            Instant instant = localDateTime.toInstant(ZoneOffset.UTC);

            ObjectLockRetention lockRetention = ObjectLockRetention.builder()
                    .mode("COMPLIANCE")
                    .retainUntilDate(instant)
                    .build();

            PutObjectRetentionRequest retentionRequest =
 PutObjectRetentionRequest.builder()
                    .bucket(bucket)
                    .key(key)
                    .bypassGovernanceRetention(true)
                    .retention(lockRetention)
                    .build();

            // To set Retention on an object, the Amazon S3 bucket must support
 object
            // locking, otherwise an exception is thrown.
            s3.putObjectRetention(retentionRequest);
            System.out.print("An object retention configuration was successfully
 placed on the object");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat PutObjectdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Unggah objek tersebut.

```
import { PutObjectCommand, S3Client } from "@aws-sdk/client-s3";

const client = new S3Client({});

export const main = async () => {
  const command = new PutObjectCommand({
    Bucket: "test-bucket",
    Key: "hello-s3.txt",
    Body: "Hello S3!",
  });

  try {
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.
- Untuk detail API, lihat PutObjectdi Referensi AWS SDK for JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun putS3Object(
    bucketName: String,
    objectKey: String,
    objectPath: String,
) {
    val metadataVal = mutableMapOf<String, String>()
    metadataVal["myVal"] = "test"

    val request =
        PutObjectRequest {
            bucket = bucketName
            key = objectKey
            metadata = metadataVal
            body = File(objectPath).asByteStream()
        }

    S3Client { region = "us-east-1" }.use { s3 ->
        val response = s3.putObject(request)
        println("Tag information is ${response.eTag}")
    }
}
```

- Untuk detail API, lihat PutObjectdi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Unggah objek ke bucket.

```
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);

        $fileName = __DIR__ . "/local-file-" . uniqid();
        try {
            $this->s3client->putObject([
                'Bucket' => $this->bucketName,
                'Key' => $fileName,
                'SourceFile' => __DIR__ . '/testfile.txt'
            ]);
            echo "Uploaded $fileName to $this->bucketName.\n";
        } catch (Exception $exception) {
            echo "Failed to upload $fileName with error: " . $exception-
>getMessage();
            exit("Please fix error with file upload before continuing.");
        }
```

- Untuk detail API, lihat PutObjectdi Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah ini mengunggah file tunggal "local-sample.txt" ke Amazon S3, membuat objek dengan kunci "sample.txt" di bucket “test-files”.

```
Write-S3Object -BucketName test-files -Key "sample.txt" -File .\local-sample.txt
```

Contoh 2: Perintah ini mengunggah file tunggal "sample.txt" ke Amazon S3, membuat objek dengan kunci "sample.txt" di bucket “test-files”. Jika parameter -Key tidak disediakan, nama file digunakan sebagai kunci objek S3.

```
Write-S3Object -BucketName test-files -File .\sample.txt
```

Contoh 3: Perintah ini mengunggah file tunggal "local-sample.txt" ke Amazon S3, membuat objek dengan kunci "prefix/to/sample.txt" di bucket “test-files”.

```
Write-S3Object -BucketName test-files -Key "prefix/to/sample.txt" -File .\local-
sample.txt
```

Contoh 4: Perintah ini mengunggah semua file di subdirektori “Scripts” ke bucket “test-files” dan menerapkan common key prefix "" untuk setiap objek. SampleScripts Setiap file yang diunggah akan memiliki kunci "SampleScripts/filename" di mana 'nama filen' bervariasi.

```
Write-S3Object -BucketName test-files -Folder .\Scripts -KeyPrefix SampleScripts\
```

Contoh 5: Perintah ini mengunggah semua file*.ps1 di direktur lokal “Scripts” ke bucket “test-files” dan menerapkan common key prefix "" ke setiap objek. SampleScripts Setiap file yang diunggah akan memiliki kunci "SampleScripts/filename.ps1" di mana 'nama filen' bervariasi.

```
Write-S3Object -BucketName test-files -Folder .\Scripts -KeyPrefix SampleScripts\
 -SearchPattern *.ps1
```

Contoh 6: Perintah ini membuat objek S3 baru yang berisi string konten tertentu dengan kunci 'sample.txt'.

```
Write-S3Object -BucketName test-files -Key "sample.txt" -Content "object
 contents"
```

Contoh 7: Perintah ini mengunggah file yang ditentukan (nama file digunakan sebagai kunci) dan menerapkan tag yang ditentukan ke objek baru.

```
Write-S3Object -BucketName test-files -File "sample.txt" -TagSet
 @{Key="key1";Value="value1"},@{Key="key2";Value="value2"}
```

Contoh 8: Perintah ini secara rekursif mengunggah folder yang ditentukan dan menerapkan tag yang ditentukan ke semua objek baru.

```
Write-S3Object -BucketName test-files -Folder . -KeyPrefix "TaggedFiles" -Recurse
 -TagSet @{Key="key1";Value="value1"},@{Key="key2";Value="value2"}
```

- Untuk detail API, lihat PutObjectdi Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    def put(self, data):
        """
        Upload data to the object.

        :param data: The data to upload. This can either be bytes or a string.
 When this
                     argument is a string, it is interpreted as a file name,
 which is
                     opened in read bytes mode.
        """
        put_data = data
        if isinstance(data, str):
            try:
```

```
                put_data = open(data, "rb")
            except IOError:
                logger.exception("Expected file name or binary data, got '%s'.",
 data)
                raise

        try:
            self.object.put(Body=put_data)
            self.object.wait_until_exists()
            logger.info(
                "Put object '%s' to bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
        except ClientError:
            logger.exception(
                "Couldn't put object '%s' to bucket '%s'.",
                self.object.key,
                self.object.bucket_name,
            )
            raise
        finally:
            if getattr(put_data, "close", None):
                put_data.close()
```

- Untuk detail API, lihat PutObjectdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Unggah file menggunakan pengunggah terkelola (Object.upload_file).

```
require "aws-sdk-s3"
```

```
# Wraps Amazon S3 object actions.
class ObjectUploadFileWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Uploads a file to an Amazon S3 object by using a managed uploader.
  #
  # @param file_path [String] The path to the file to upload.
  # @return [Boolean] True when the file is uploaded; otherwise false.
  def upload_file(file_path)
    @object.upload_file(file_path)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't upload file #{file_path} to #{@object.key}. Here's why:
 #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-uploaded-file"
  file_path = "object_upload_file.rb"

  wrapper = ObjectUploadFileWrapper.new(Aws::S3::Object.new(bucket_name,
 object_key))
  return unless wrapper.upload_file(file_path)

  puts "File #{file_path} successfully uploaded to #{bucket_name}:#{object_key}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Unggah file menggunakan Object.put.

```
require "aws-sdk-s3"
```

```
# Wraps Amazon S3 object actions.
class ObjectPutWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  def put_object(source_file_path)
    File.open(source_file_path, "rb") do |file|
      @object.put(body: file)
    end
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't put #{source_file_path} to #{object.key}. Here's why:
 #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object-key"
  file_path = "my-local-file.txt"

  wrapper = ObjectPutWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
  success = wrapper.put_object(file_path)
  return unless success

  puts "Put file #{file_path} into #{object_key} in #{bucket_name}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Unggah file menggunakan Object.put dan tambahkan enkripsi di sisi server.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
```

```
class ObjectPutSseWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  def put_object_encrypted(object_content, encryption)
    @object.put(body: object_content, server_side_encryption: encryption)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't put your content to #{object.key}. Here's why: #{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-encrypted-content"
  object_content = "This is my super-secret content."
  encryption = "AES256"

  wrapper = ObjectPutSseWrapper.new(Aws::S3::Object.new(bucket_name,
 object_content))
  return unless wrapper.put_object_encrypted(object_content, encryption)

  puts "Put your content into #{bucket_name}:#{object_key} and encrypted it with
 #{encryption}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk detail API, lihat PutObjectdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
pub async fn upload_object(
    client: &Client,
    bucket_name: &str,
    file_name: &str,
    key: &str,
) -> Result<PutObjectOutput, SdkError<PutObjectError>> {
    let body = ByteStream::from_path(Path::new(file_name)).await;
    client
        .put_object()
        .bucket(bucket_name)
        .key(key)
        .body(body.unwrap())
        .send()
        .await
}
```

- Untuk detail API, lihat PutObjectreferensi AWS SDK for Rust API.

SAP ABAP

SDK untuk SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    "Get contents of file from application server."
    DATA lv_body TYPE xstring.
    OPEN DATASET iv_file_name FOR INPUT IN BINARY MODE.
    READ DATASET iv_file_name INTO lv_body.
    CLOSE DATASET iv_file_name.

    "Upload/put an object to an S3 bucket."
    TRY.
        lo_s3->putobject(
            iv_bucket = iv_bucket_name
            iv_key = iv_file_name
            iv_body = lv_body
        ).
        MESSAGE 'Object uploaded to S3 bucket.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
    ENDTRY.
```

- Untuk detail API, lihat PutObjectdi AWS SDK untuk referensi SAP ABAP API.

Swift

SDK untuk Swift

> **Note**
>
> Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Unggah file dari penyimpanan lokal ke bucket.

```
    public func uploadFile(bucket: String, key: String, file: String) async
throws {
        let fileUrl = URL(fileURLWithPath: file)
        let fileData = try Data(contentsOf: fileUrl)
        let dataStream = ByteStream.from(data: fileData)

        let input = PutObjectInput(
            body: dataStream,
            bucket: bucket,
            key: key
        )
        _ = try await client.putObject(input: input)
    }
```

Unggah konten objek Data Swift ke bucket.

```
    public func createFile(bucket: String, key: String, withData data: Data)
async throws {
        let dataStream = ByteStream.from(data: data)

        let input = PutObjectInput(
            body: dataStream,
            bucket: bucket,
            key: key
        )
        _ = try await client.putObject(input: input)
    }
```

- Untuk detail API, lihat referensi PutObject AWSSDK untuk Swift API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutObjectAcl`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutObjectAcl`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Mengelola daftar kontrol akses (ACL)

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
bool AwsDoc::S3::putObjectAcl(const Aws::String &bucketName, const Aws::String
 &objectKey, const Aws::String &ownerID,
                              const Aws::String &granteePermission, const
 Aws::String &granteeType,
                              const Aws::String &granteeID, const Aws::String
 &granteeEmailAddress,
                              const Aws::String &granteeURI, const
 Aws::S3::S3ClientConfiguration &clientConfig) {
    Aws::S3::S3Client s3Client(clientConfig);

    Aws::S3::Model::Owner owner;
    owner.SetID(ownerID);

    Aws::S3::Model::Grantee grantee;
    grantee.SetType(setGranteeType(granteeType));

    if (!granteeEmailAddress.empty()) {
        grantee.SetEmailAddress(granteeEmailAddress);
    }

    if (!granteeID.empty()) {
        grantee.SetID(granteeID);
    }

    if (!granteeURI.empty()) {
        grantee.SetURI(granteeURI);
```

```
    }

    Aws::S3::Model::Grant grant;
    grant.SetGrantee(grantee);
    grant.SetPermission(setGranteePermission(granteePermission));

    Aws::Vector<Aws::S3::Model::Grant> grants;
    grants.push_back(grant);

    Aws::S3::Model::AccessControlPolicy acp;
    acp.SetOwner(owner);
    acp.SetGrants(grants);

    Aws::S3::Model::PutObjectAclRequest request;
    request.SetAccessControlPolicy(acp);
    request.SetBucket(bucketName);
    request.SetKey(objectKey);

    Aws::S3::Model::PutObjectAclOutcome outcome =
            s3Client.PutObjectAcl(request);

    if (!outcome.IsSuccess()) {
        auto error = outcome.GetError();
        std::cerr << "Error: putObjectAcl: " << error.GetExceptionName()
                  << " - " << error.GetMessage() << std::endl;
    } else {
        std::cout << "Successfully added an ACL to the object '" << objectKey
                  << "' in the bucket '" << bucketName << "'." << std::endl;
    }

    return outcome.IsSuccess();
}

//! Routine which converts a human-readable string to a built-in type
 enumeration.
/*!
 \param access: Human readable string.
 \return Permission: Permission enumeration.
*/
Aws::S3::Model::Permission setGranteePermission(const Aws::String &access) {
    if (access == "FULL_CONTROL")
        return Aws::S3::Model::Permission::FULL_CONTROL;
    if (access == "WRITE")
        return Aws::S3::Model::Permission::WRITE;
```

```
    if (access == "READ")
        return Aws::S3::Model::Permission::READ;
    if (access == "WRITE_ACP")
        return Aws::S3::Model::Permission::WRITE_ACP;
    if (access == "READ_ACP")
        return Aws::S3::Model::Permission::READ_ACP;
    return Aws::S3::Model::Permission::NOT_SET;
}

//! Routine which converts a human-readable string to a built-in type
 enumeration.
/*!
 \param type: Human readable string.
 \return Type: Type enumeration.
*/
Aws::S3::Model::Type setGranteeType(const Aws::String &type) {
    if (type == "Amazon customer by email")
        return Aws::S3::Model::Type::AmazonCustomerByEmail;
    if (type == "Canonical user")
        return Aws::S3::Model::Type::CanonicalUser;
    if (type == "Group")
        return Aws::S3::Model::Type::Group;
    return Aws::S3::Model::Type::NOT_SET;
}
```

- Untuk detail API, lihat PutObjectAcldi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Perintah berikut memberikan `full control` kepada dua AWS pengguna (user1@example.com dan user2@example.com) dan `read` izin untuk semua orang:

```
aws s3api put-object-acl --bucket MyBucket --key file.txt --grant-full-control
 emailaddress=user1@example.com,emailaddress=user2@example.com --grant-read
 uri=http://acs.amazonaws.com/groups/global/AllUsers
```

Lihat http://docs.aws.amazon.com/AmazonS3/latest/API/RESTBucketPUTacl.html untuk detail tentang ACL kustom (perintah s3api ACL, sepertiput`-object-acl`, menggunakan notasi argumen singkatan yang sama).

- Untuk detail API, lihat PutObjectAcldi Referensi AWS CLI Perintah.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class ObjectWrapper:
    """Encapsulates S3 object actions."""

    def __init__(self, s3_object):
        """
        :param s3_object: A Boto3 Object resource. This is a high-level resource
 in Boto3
                          that wraps object actions in a class-like structure.
        """
        self.object = s3_object
        self.key = self.object.key


    def put_acl(self, email):
        """
        Applies an ACL to the object that grants read access to an AWS user
 identified
        by email address.

        :param email: The email address of the user to grant access.
        """
        try:
            acl = self.object.Acl()
            # Putting an ACL overwrites the existing ACL, so append new grants
            # if you want to preserve existing grants.
            grants = acl.grants if acl.grants else []
```

```
            grants.append(
                {
                    "Grantee": {"Type": "AmazonCustomerByEmail", "EmailAddress":
email},
                    "Permission": "READ",
                }
            )
            acl.put(AccessControlPolicy={"Grants": grants, "Owner": acl.owner})
            logger.info("Granted read access to %s.", email)
        except ClientError:
            logger.exception("Couldn't add ACL to object '%s'.", self.object.key)
            raise
```

- Untuk detail API, lihat PutObjectAcldi AWS SDK for Python (Boto3) Referensi API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutObjectLegalHold`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutObjectLegalHold`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kunci objek Amazon S3

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// Set or modify a legal hold on an object in an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The key of the object.</param>
    /// <param name="holdStatus">The On or Off status for the legal hold.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ModifyObjectLegalHold(string bucketName,
        string objectKey, ObjectLockLegalHoldStatus holdStatus)
    {
        try
        {
            var request = new PutObjectLegalHoldRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
                LegalHold = new ObjectLockLegalHold()
                {
                    Status = holdStatus
                }
            };

            var response = await _amazonS3.PutObjectLegalHoldAsync(request);
            Console.WriteLine($"\tModified legal hold for {objectKey} in
{bucketName}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tError modifying legal hold: '{ex.Message}'");
            return false;
        }
    }
```

- Untuk detail API, lihat PutObjectLegalHolddi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk menerapkan Penahanan Hukum pada suatu objek

`put-object-legal-hold`Contoh berikut menetapkan Penahanan Hukum pada objek`doc1.rtf`.

```
aws s3api put-object-legal-hold \
    --bucket my-bucket-with-object-lock \
    --key doc1.rtf \
    --legal-hold Status=ON
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat PutObjectLegalHolddi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// S3Actions wraps S3 service actions.
type S3Actions struct {
 S3Client  *s3.Client
 S3Manager *manager.Uploader
}



// PutObjectLegalHold sets the legal hold configuration for an S3 object.
func (actor S3Actions) PutObjectLegalHold(ctx context.Context, bucket string, key
 string, versionId string, legalHoldStatus types.ObjectLockLegalHoldStatus) error
 {
 input := &s3.PutObjectLegalHoldInput{
  Bucket: aws.String(bucket),
  Key:    aws.String(key),
  LegalHold: &types.ObjectLockLegalHold{
   Status: legalHoldStatus,
  },
```

```
 }
 if versionId != "" {
  input.VersionId = aws.String(versionId)
 }

 _, err := actor.S3Client.PutObjectLegalHold(ctx, input)
 if err != nil {
  var noKey *types.NoSuchKey
  if errors.As(err, &noKey) {
   log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
   err = noKey
  }
 }

 return err
}
```

- Untuk detail API, lihat PutObjectLegalHolddi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    // Set or modify a legal hold on an object in an S3 bucket.
    public void modifyObjectLegalHold(String bucketName, String objectKey,
 boolean legalHoldOn) {
        ObjectLockLegalHold legalHold ;
        if (legalHoldOn) {
            legalHold = ObjectLockLegalHold.builder()
                .status(ObjectLockLegalHoldStatus.ON)
                .build();
        } else {
            legalHold = ObjectLockLegalHold.builder()
                .status(ObjectLockLegalHoldStatus.OFF)
```

```
                .build();
        }

        PutObjectLegalHoldRequest legalHoldRequest =
 PutObjectLegalHoldRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .legalHold(legalHold)
            .build();

        getClient().putObjectLegalHold(legalHoldRequest) ;
        System.out.println("Modified legal hold for "+ objectKey +" in
 "+bucketName +".");
    }
```

- Untuk detail API, lihat PutObjectLegalHolddi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { fileURLToPath } from "url";
import { PutObjectLegalHoldCommand, S3Client } from "@aws-sdk/client-s3";

/**
 * @param {S3Client} client
 * @param {string} bucketName
 * @param {string} objectKey
 */
export const main = async (client, bucketName, objectKey) => {
  const command = new PutObjectLegalHoldCommand({
    Bucket: bucketName,
    Key: objectKey,
```

```
    LegalHold: {
      // Set the status to 'ON' to place a legal hold on the object.
      // Set the status to 'OFF' to remove the legal hold.
      Status: "ON",
    },
    // Optionally, you can provide additional parameters
    // ChecksumAlgorithm: "ALGORITHM",
    // ContentMD5: "MD5_HASH",
    // ExpectedBucketOwner: "ACCOUNT_ID",
    // RequestPayer: "requester",
    // VersionId: "OBJECT_VERSION_ID",
  });

  try {
    const response = await client.send(command);
    console.log(
      `Object legal hold status: ${response.$metadata.httpStatusCode}`,
    );
  } catch (err) {
    console.error(err);
  }
};

// Invoke main function if this file was run directly.
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  main(new S3Client(), "BUCKET_NAME", "OBJECT_KEY");
}
```

- Untuk detail API, lihat PutObjectLegalHolddi Referensi AWS SDK for JavaScript API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutObjectLockConfiguration`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutObjectLockConfiguration`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kunci objek Amazon S3

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Atur konfigurasi kunci objek dari ember.

```
    /// <summary>
    /// Enable object lock on an existing bucket.
    /// </summary>
    /// <param name="bucketName">The name of the bucket to modify.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> EnableObjectLockOnBucket(string bucketName)
    {
        try
        {
            // First, enable Versioning on the bucket.
            await _amazonS3.PutBucketVersioningAsync(new
PutBucketVersioningRequest()
            {
                BucketName = bucketName,
                VersioningConfig = new S3BucketVersioningConfig()
                {
                    EnableMfaDelete = false,
                    Status = VersionStatus.Enabled
                }
            });

            var request = new PutObjectLockConfigurationRequest()
            {
                BucketName = bucketName,
                ObjectLockConfiguration = new ObjectLockConfiguration()
                {
                    ObjectLockEnabled = new ObjectLockEnabled("Enabled"),
                },
```

```
            };

            var response = await
_amazonS3.PutObjectLockConfigurationAsync(request);
            Console.WriteLine($"\tAdded an object lock policy to bucket
{bucketName}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"Error modifying object lock: '{ex.Message}'");
            return false;
        }
    }
```

Setel periode retensi default bucket.

```
    /// <summary>
    /// Set or modify a retention period on an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket to modify.</param>
    /// <param name="retention">The retention mode.</param>
    /// <param name="retainUntilDate">The date for retention until.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ModifyBucketDefaultRetention(string bucketName, bool
enableObjectLock, ObjectLockRetentionMode retention, DateTime retainUntilDate)
    {
        var enabledString = enableObjectLock ? "Enabled" : "Disabled";
        var timeDifference = retainUntilDate.Subtract(DateTime.Now);
        try
        {
            // First, enable Versioning on the bucket.
            await _amazonS3.PutBucketVersioningAsync(new
PutBucketVersioningRequest()
            {
                BucketName = bucketName,
                VersioningConfig = new S3BucketVersioningConfig()
                {
                    EnableMfaDelete = false,
                    Status = VersionStatus.Enabled
                }
            });
```

```
            var request = new PutObjectLockConfigurationRequest()
            {
                BucketName = bucketName,
                ObjectLockConfiguration = new ObjectLockConfiguration()
                {
                    ObjectLockEnabled = new ObjectLockEnabled(enabledString),
                    Rule = new ObjectLockRule()
                    {
                        DefaultRetention = new DefaultRetention()
                        {
                            Mode = retention,
                            Days = timeDifference.Days // Can be specified in
days or years but not both.
                        }
                    }
                }
            };

            var response = await
_amazonS3.PutObjectLockConfigurationAsync(request);
            Console.WriteLine($"\tAdded a default retention to bucket
{bucketName}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tError modifying object lock: '{ex.Message}'");
            return false;
        }
    }
```

- Untuk detail API, lihat PutObjectLockConfigurationdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk mengatur konfigurasi kunci objek pada ember

`put-object-lock-configuration`Contoh berikut menetapkan kunci objek 50 hari pada bucket yang ditentukan.

```
aws s3api put-object-lock-configuration \
    --bucket my-bucket-with-object-lock \
    --object-lock-configuration '{ "ObjectLockEnabled": "Enabled", "Rule":
 { "DefaultRetention": { "Mode": "COMPLIANCE", "Days": 50 }}}'
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat PutObjectLockConfigurationdi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Atur konfigurasi kunci objek dari ember.

```
// S3Actions wraps S3 service actions.
type S3Actions struct {
 S3Client  *s3.Client
 S3Manager *manager.Uploader
}



// EnableObjectLockOnBucket enables object locking on an existing bucket.
func (actor S3Actions) EnableObjectLockOnBucket(ctx context.Context, bucket
 string) error {
 // Versioning must be enabled on the bucket before object locking is enabled.
 verInput := &s3.PutBucketVersioningInput{
  Bucket: aws.String(bucket),
  VersioningConfiguration: &types.VersioningConfiguration{
   MFADelete: types.MFADeleteDisabled,
   Status:    types.BucketVersioningStatusEnabled,
  },
 }
 _, err := actor.S3Client.PutBucketVersioning(ctx, verInput)
```

```
 if err != nil {
  var noBucket *types.NoSuchBucket
  if errors.As(err, &noBucket) {
   log.Printf("Bucket %s does not exist.\n", bucket)
   err = noBucket
  }
  return err
 }

 input := &s3.PutObjectLockConfigurationInput{
  Bucket: aws.String(bucket),
  ObjectLockConfiguration: &types.ObjectLockConfiguration{
   ObjectLockEnabled: types.ObjectLockEnabledEnabled,
  },
 }
 _, err = actor.S3Client.PutObjectLockConfiguration(ctx, input)
 if err != nil {
  var noBucket *types.NoSuchBucket
  if errors.As(err, &noBucket) {
   log.Printf("Bucket %s does not exist.\n", bucket)
   err = noBucket
  }
 }

 return err
}
```

Setel periode retensi default bucket.

```
// S3Actions wraps S3 service actions.
type S3Actions struct {
 S3Client  *s3.Client
 S3Manager *manager.Uploader
}



// ModifyDefaultBucketRetention modifies the default retention period of an
 existing bucket.
func (actor S3Actions) ModifyDefaultBucketRetention(
```

```
 ctx context.Context, bucket string, lockMode types.ObjectLockEnabled,
 retentionPeriod int32, retentionMode types.ObjectLockRetentionMode) error {

 input := &s3.PutObjectLockConfigurationInput{
  Bucket: aws.String(bucket),
  ObjectLockConfiguration: &types.ObjectLockConfiguration{
   ObjectLockEnabled: lockMode,
   Rule: &types.ObjectLockRule{
    DefaultRetention: &types.DefaultRetention{
     Days: aws.Int32(retentionPeriod),
     Mode: retentionMode,
    },
   },
  },
 }
 _, err := actor.S3Client.PutObjectLockConfiguration(ctx, input)
 if err != nil {
  var noBucket *types.NoSuchBucket
  if errors.As(err, &noBucket) {
   log.Printf("Bucket %s does not exist.\n", bucket)
   err = noBucket
  }
 }

 return err
}
```

- Untuk detail API, lihat PutObjectLockConfigurationdi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Atur konfigurasi kunci objek dari ember.

```
   // Enable object lock on an existing bucket.
   public void enableObjectLockOnBucket(String bucketName) {
       try {
           VersioningConfiguration versioningConfiguration =
VersioningConfiguration.builder()
               .status(BucketVersioningStatus.ENABLED)
               .build();

           PutBucketVersioningRequest putBucketVersioningRequest =
PutBucketVersioningRequest.builder()
               .bucket(bucketName)
               .versioningConfiguration(versioningConfiguration)
               .build();

           // Enable versioning on the bucket.
           getClient().putBucketVersioning(putBucketVersioningRequest);
           PutObjectLockConfigurationRequest request =
PutObjectLockConfigurationRequest.builder()
               .bucket(bucketName)
               .objectLockConfiguration(ObjectLockConfiguration.builder()
                   .objectLockEnabled(ObjectLockEnabled.ENABLED)
                   .build())
               .build();

           getClient().putObjectLockConfiguration(request);
           System.out.println("Successfully enabled object lock on
"+bucketName);

       } catch (S3Exception ex) {
           System.out.println("Error modifying object lock: '" + ex.getMessage()
+ "'");
       }
   }
```

Setel periode retensi default bucket.

```
   // Set or modify a retention period on an S3 bucket.
   public void modifyBucketDefaultRetention(String bucketName) {
       VersioningConfiguration versioningConfiguration =
VersioningConfiguration.builder()
           .mfaDelete(MFADelete.DISABLED)
           .status(BucketVersioningStatus.ENABLED)
```

```
            .build();

        PutBucketVersioningRequest versioningRequest =
PutBucketVersioningRequest.builder()
            .bucket(bucketName)
            .versioningConfiguration(versioningConfiguration)
            .build();

        getClient().putBucketVersioning(versioningRequest);
        DefaultRetention rention = DefaultRetention.builder()
            .days(1)
            .mode(ObjectLockRetentionMode.GOVERNANCE)
            .build();

        ObjectLockRule lockRule = ObjectLockRule.builder()
            .defaultRetention(rention)
            .build();

        ObjectLockConfiguration objectLockConfiguration =
ObjectLockConfiguration.builder()
            .objectLockEnabled(ObjectLockEnabled.ENABLED)
            .rule(lockRule)
            .build();

        PutObjectLockConfigurationRequest putObjectLockConfigurationRequest =
PutObjectLockConfigurationRequest.builder()
            .bucket(bucketName)
            .objectLockConfiguration(objectLockConfiguration)
            .build();


getClient().putObjectLockConfiguration(putObjectLockConfigurationRequest) ;
        System.out.println("Added a default retention to bucket "+bucketName
+".");
    }
```

- Untuk detail API, lihat PutObjectLockConfigurationdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Atur konfigurasi kunci objek dari ember.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { fileURLToPath } from "url";
import {
  PutObjectLockConfigurationCommand,
  S3Client,
} from "@aws-sdk/client-s3";

/**
 * @param {S3Client} client
 * @param {string} bucketName
 */
export const main = async (client, bucketName) => {
  const command = new PutObjectLockConfigurationCommand({
    Bucket: bucketName,
    // The Object Lock configuration that you want to apply to the specified
 bucket.
    ObjectLockConfiguration: {
      ObjectLockEnabled: "Enabled",
    },
    // Optionally, you can provide additional parameters
    // ExpectedBucketOwner: "ACCOUNT_ID",
    // RequestPayer: "requester",
    // Token: "OPTIONAL_TOKEN",
  });

  try {
    const response = await client.send(command);
    console.log(
      `Object Lock Configuration updated: ${response.$metadata.httpStatusCode}`,
    );
```

```
  } catch (err) {
    console.error(err);
  }
};

// Invoke main function if this file was run directly.
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  main(new S3Client(), "BUCKET_NAME");
}
```

Setel periode retensi default bucket.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { fileURLToPath } from "url";
import {
  PutObjectLockConfigurationCommand,
  S3Client,
} from "@aws-sdk/client-s3";

/**
 * @param {S3Client} client
 * @param {string} bucketName
 */
export const main = async (client, bucketName) => {
  const command = new PutObjectLockConfigurationCommand({
    Bucket: bucketName,
    // The Object Lock configuration that you want to apply to the specified
 bucket.
    ObjectLockConfiguration: {
      ObjectLockEnabled: "Enabled",
      Rule: {
        DefaultRetention: {
          Mode: "GOVERNANCE",
          Years: 3,
        },
      },
    },
    // Optionally, you can provide additional parameters
    // ExpectedBucketOwner: "ACCOUNT_ID",
    // RequestPayer: "requester",
    // Token: "OPTIONAL_TOKEN",
```

```
  });

  try {
    const response = await client.send(command);
    console.log(
      `Default Object Lock Configuration updated: ${response.
$metadata.httpStatusCode}`,
    );
  } catch (err) {
    console.error(err);
  }
};

// Invoke main function if this file was run directly.
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  main(new S3Client(), "BUCKET_NAME");
}
```

- Untuk detail API, lihat PutObjectLockConfigurationdi Referensi AWS SDK for JavaScript API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`PutObjectRetention`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`PutObjectRetention`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Kunci objek Amazon S3

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// Set or modify a retention period on an object in an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The key of the object.</param>
    /// <param name="retention">The retention mode.</param>
    /// <param name="retainUntilDate">The date retention expires.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ModifyObjectRetentionPeriod(string bucketName,
        string objectKey, ObjectLockRetentionMode retention, DateTime
retainUntilDate)
    {
        try
        {
            var request = new PutObjectRetentionRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
                Retention = new ObjectLockRetention()
                {
                    Mode = retention,
                    RetainUntilDate = retainUntilDate
                }
            };

            var response = await _amazonS3.PutObjectRetentionAsync(request);
            Console.WriteLine($"\tSet retention for {objectKey} in {bucketName}
until {retainUntilDate:d}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
```

```
            Console.WriteLine($"\tError modifying retention period:
 '{ex.Message}'");
            return false;
        }
    }
```

- Untuk detail API, lihat PutObjectRetentionDi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk mengatur konfigurasi retensi objek untuk objek

`put-object-retention`Contoh berikut menetapkan konfigurasi retensi objek untuk objek yang ditentukan hingga 2025-01-01.

```
aws s3api put-object-retention \
    --bucket my-bucket-with-object-lock \
    --key doc1.rtf \
    --retention '{ "Mode": "GOVERNANCE", "RetainUntilDate":
 "2025-01-01T00:00:00" }'
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat PutObjectRetentionDi Referensi AWS CLI Perintah.

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// S3Actions wraps S3 service actions.
```

```
type S3Actions struct {
 S3Client  *s3.Client
 S3Manager *manager.Uploader
}



// PutObjectRetention sets the object retention configuration for an S3 object.
func (actor S3Actions) PutObjectRetention(ctx context.Context, bucket string, key
 string, retentionMode types.ObjectLockRetentionMode, retentionPeriodDays int32)
 error {
 input := &s3.PutObjectRetentionInput{
  Bucket: aws.String(bucket),
  Key:    aws.String(key),
  Retention: &types.ObjectLockRetention{
   Mode:            retentionMode,
   RetainUntilDate: aws.Time(time.Now().AddDate(0, 0, int(retentionPeriodDays))),
  },
  BypassGovernanceRetention: aws.Bool(true),
 }

 _, err := actor.S3Client.PutObjectRetention(ctx, input)
 if err != nil {
  var noKey *types.NoSuchKey
  if errors.As(err, &noKey) {
   log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
   err = noKey
  }
 }

 return err
}
```

- Untuk detail API, lihat PutObjectRetentiondi Referensi AWS SDK for Go API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
   // Set or modify a retention period on an object in an S3 bucket.
   public void modifyObjectRetentionPeriod(String bucketName, String objectKey)
{
       // Calculate the instant one day from now.
       Instant futureInstant = Instant.now().plus(1, ChronoUnit.DAYS);

       // Convert the Instant to a ZonedDateTime object with a specific time
zone.
       ZonedDateTime zonedDateTime =
futureInstant.atZone(ZoneId.systemDefault());

       // Define a formatter for human-readable output.
       DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd
HH:mm:ss");

       // Format the ZonedDateTime object to a human-readable date string.
       String humanReadableDate = formatter.format(zonedDateTime);

       // Print the formatted date string.
       System.out.println("Formatted Date: " + humanReadableDate);
       ObjectLockRetention retention = ObjectLockRetention.builder()
           .mode(ObjectLockRetentionMode.GOVERNANCE)
           .retainUntilDate(futureInstant)
           .build();

       PutObjectRetentionRequest retentionRequest =
PutObjectRetentionRequest.builder()
           .bucket(bucketName)
           .key(objectKey)
           .retention(retention)
           .build();
```

```
        getClient().putObjectRetention(retentionRequest);
        System.out.println("Set retention for "+objectKey +" in " +bucketName +"
 until "+ humanReadableDate +".");
    }
```

- Untuk detail API, lihat PutObjectRetentiondi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { fileURLToPath } from "url";
import { PutObjectRetentionCommand, S3Client } from "@aws-sdk/client-s3";

/**
 * @param {S3Client} client
 * @param {string} bucketName
 * @param {string} objectKey
 */
export const main = async (client, bucketName, objectKey) => {
  const command = new PutObjectRetentionCommand({
    Bucket: bucketName,
    Key: objectKey,
    BypassGovernanceRetention: false,
    // ChecksumAlgorithm: "ALGORITHM",
    // ContentMD5: "MD5_HASH",
    // ExpectedBucketOwner: "ACCOUNT_ID",
    // RequestPayer: "requester",
    Retention: {
      Mode: "GOVERNANCE", // or "COMPLIANCE"
      RetainUntilDate: new Date(new Date().getTime() + 24 * 60 * 60 * 1000),
    },
    // VersionId: "OBJECT_VERSION_ID",
```

```
  });

  try {
    const response = await client.send(command);
    console.log(
      `Object Retention settings updated: ${response.$metadata.httpStatusCode}`,
    );
  } catch (err) {
    console.error(err);
  }
};

// Invoke main function if this file was run directly.
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  main(new S3Client(), "BUCKET_NAME", "OBJECT_KEY");
}
```

- Untuk detail API, lihat PutObjectRetentiondi Referensi AWS SDK for JavaScript API.

PowerShell

Alat untuk PowerShell

Contoh 1: Perintah mengaktifkan mode retensi tata kelola hingga tanggal '31 Des 2019 00:00:00 'untuk objek' testfile.txt 'di bucket S3 yang diberikan.

```
Write-S3ObjectRetention -BucketName 's3buckettesting' -Key 'testfile.txt' -
Retention_Mode GOVERNANCE -Retention_RetainUntilDate "2019-12-31T00:00:00"
```

- Untuk detail API, lihat PutObjectRetentiondi Referensi AWS Tools for PowerShell Cmdlet.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`RestoreObject`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`RestoreObject`.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to restore an archived object in an Amazon
/// Simple Storage Service (Amazon S3) bucket.
/// </summary>
public class RestoreArchivedObject
{
    public static void Main()
    {
        string bucketName = "doc-example-bucket";
        string objectKey = "archived-object.txt";

        // Specify your bucket region (an example region is shown).
        RegionEndpoint bucketRegion = RegionEndpoint.USWest2;

        IAmazonS3 client = new AmazonS3Client(bucketRegion);
        RestoreObjectAsync(client, bucketName, objectKey).Wait();
    }

    /// <summary>
    /// This method restores an archived object from an Amazon S3 bucket.
    /// </summary>
    /// <param name="client">The initialized Amazon S3 client object used to
call
    /// RestoreObjectAsync.</param>
    /// <param name="bucketName">A string representing the name of the
    /// bucket where the object was located before it was archived.</param>
```

```
        /// <param name="objectKey">A string representing the name of the
        /// archived object to restore.</param>
        public static async Task RestoreObjectAsync(IAmazonS3 client, string
bucketName, string objectKey)
        {
            try
            {
                var restoreRequest = new RestoreObjectRequest
                {
                    BucketName = bucketName,
                    Key = objectKey,
                    Days = 2,
                };
                RestoreObjectResponse response = await
client.RestoreObjectAsync(restoreRequest);

                // Check the status of the restoration.
                await CheckRestorationStatusAsync(client, bucketName, objectKey);
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine($"Error: {amazonS3Exception.Message}");
            }
        }

        /// <summary>
        /// This method retrieves the status of the object's restoration.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// GetObjectMetadataAsync.</param>
        /// <param name="bucketName">A string representing the name of the Amazon
        /// S3 bucket which contains the archived object.</param>
        /// <param name="objectKey">A string representing the name of the
        /// archived object you want to restore.</param>
        public static async Task CheckRestorationStatusAsync(IAmazonS3 client,
string bucketName, string objectKey)
        {
            GetObjectMetadataRequest metadataRequest = new
GetObjectMetadataRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
            };
```

```
            GetObjectMetadataResponse response = await
 client.GetObjectMetadataAsync(metadataRequest);

            var restStatus = response.RestoreInProgress ? "in-progress" :
 "finished or failed";
            Console.WriteLine($"Restoration status: {restStatus}");
        }
    }
```

- Untuk detail API, lihat RestoreObjectdi Referensi AWS SDK for .NET API.

CLI

AWS CLI

Untuk membuat permintaan pemulihan untuk objek

`restore-object`Contoh berikut mengembalikan objek Amazon S3 Glacier yang ditentukan untuk bucket selama 10 hari. `my-glacier-bucket`

```
aws s3api restore-object \
    --bucket my-glacier-bucket \
    --key doc1.rtf \
    --restore-request Days=10
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat RestoreObjectdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.RestoreRequest;
import software.amazon.awssdk.services.s3.model.GlacierJobParameters;
import software.amazon.awssdk.services.s3.model.RestoreObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.Tier;

/*
 *  For more information about restoring an object, see "Restoring an archived
 object" at
 *  https://docs.aws.amazon.com/AmazonS3/latest/userguide/restoring-objects.html
 *
 *  Before running this Java V2 code example, set up your development
 environment, including your credentials.
 *
 *  For more information, see the following documentation topic:
 *
 *  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class RestoreObject {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <bucketName> <keyName> <expectedBucketOwner>

                Where:
                    bucketName - The Amazon S3 bucket name.\s
                    keyName - The key name of an object with a Storage class
 value of Glacier.\s
                    expectedBucketOwner - The account that owns the bucket (you
 can obtain this value from the AWS Management Console).\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String keyName = args[1];
```

```
        String expectedBucketOwner = args[2];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        restoreS3Object(s3, bucketName, keyName, expectedBucketOwner);
        s3.close();
    }

    public static void restoreS3Object(S3Client s3, String bucketName, String
 keyName, String expectedBucketOwner) {
        try {
            RestoreRequest restoreRequest = RestoreRequest.builder()
                    .days(10)

 .glacierJobParameters(GlacierJobParameters.builder().tier(Tier.STANDARD).build())
                    .build();

            RestoreObjectRequest objectRequest = RestoreObjectRequest.builder()
                    .expectedBucketOwner(expectedBucketOwner)
                    .bucket(bucketName)
                    .key(keyName)
                    .restoreRequest(restoreRequest)
                    .build();

            s3.restoreObject(objectRequest);

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat RestoreObjectdi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan `SelectObjectContent` dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`SelectObjectContent`.

CLI

AWS CLI

Untuk memfilter konten objek Amazon S3 berdasarkan pernyataan SQL

`select-object-content`Contoh berikut menyaring objek `my-data-file.csv` dengan pernyataan SQL tertentu dan mengirimkan output ke file.

```
aws s3api select-object-content \
    --bucket my-bucket \
    --key my-data-file.csv \
    --expression "select * from s3object limit 100" \
    --expression-type 'SQL' \
    --input-serialization '{"CSV": {}, "CompressionType": "NONE"}' \
    --output-serialization '{"CSV": {}}' "output.csv"
```

Perintah ini tidak menghasilkan output.

- Untuk detail API, lihat SelectObjectContentdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Contoh berikut menunjukkan query menggunakan objek JSON. Contoh lengkap juga menunjukkan penggunaan objek CSV.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.async.AsyncRequestBody;
```

```
import software.amazon.awssdk.core.async.BlockingInputStreamAsyncRequestBody;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.services.s3.S3AsyncClient;
import software.amazon.awssdk.services.s3.model.CSVInput;
import software.amazon.awssdk.services.s3.model.CSVOutput;
import software.amazon.awssdk.services.s3.model.CompressionType;
import software.amazon.awssdk.services.s3.model.ExpressionType;
import software.amazon.awssdk.services.s3.model.FileHeaderInfo;
import software.amazon.awssdk.services.s3.model.InputSerialization;
import software.amazon.awssdk.services.s3.model.JSONInput;
import software.amazon.awssdk.services.s3.model.JSONOutput;
import software.amazon.awssdk.services.s3.model.JSONType;
import software.amazon.awssdk.services.s3.model.ObjectIdentifier;
import software.amazon.awssdk.services.s3.model.OutputSerialization;
import software.amazon.awssdk.services.s3.model.Progress;
import software.amazon.awssdk.services.s3.model.PutObjectResponse;
import software.amazon.awssdk.services.s3.model.SelectObjectContentRequest;
import
 software.amazon.awssdk.services.s3.model.SelectObjectContentResponseHandler;
import software.amazon.awssdk.services.s3.model.Stats;

import java.io.IOException;
import java.net.URL;
import java.util.ArrayList;
import java.util.List;
import java.util.UUID;
import java.util.concurrent.CompletableFuture;

public class SelectObjectContentExample {
    static final Logger logger =
 LoggerFactory.getLogger(SelectObjectContentExample.class);
    static final String BUCKET_NAME = "select-object-content-" +
 UUID.randomUUID();
    static final S3AsyncClient s3AsyncClient = S3AsyncClient.create();
    static String FILE_CSV = "csv";
    static String FILE_JSON = "json";
    static String URL_CSV = "https://raw.githubusercontent.com/mledoze/countries/
master/dist/countries.csv";
    static String URL_JSON = "https://raw.githubusercontent.com/mledoze/
countries/master/dist/countries.json";

    public static void main(String[] args) {
        SelectObjectContentExample selectObjectContentExample = new
 SelectObjectContentExample();
```

```
        try {
            SelectObjectContentExample.setUp();
            selectObjectContentExample.runSelectObjectContentMethodForJSON();
            selectObjectContentExample.runSelectObjectContentMethodForCSV();
        } catch (SdkException e) {
            logger.error(e.getMessage(), e);
            System.exit(1);
        } finally {
            SelectObjectContentExample.tearDown();
        }
    }

    EventStreamInfo runSelectObjectContentMethodForJSON() {
        // Set up request parameters.
        final String queryExpression = "select * from s3object[*][*] c where
c.area < 350000";
        final String fileType = FILE_JSON;

        InputSerialization inputSerialization = InputSerialization.builder()
                .json(JSONInput.builder().type(JSONType.DOCUMENT).build())
                .compressionType(CompressionType.NONE)
                .build();

        OutputSerialization outputSerialization = OutputSerialization.builder()
                .json(JSONOutput.builder().recordDelimiter(null).build())
                .build();

        // Build the SelectObjectContentRequest.
        SelectObjectContentRequest select = SelectObjectContentRequest.builder()
                .bucket(BUCKET_NAME)
                .key(FILE_JSON)
                .expression(queryExpression)
                .expressionType(ExpressionType.SQL)
                .inputSerialization(inputSerialization)
                .outputSerialization(outputSerialization)
                .build();

        EventStreamInfo eventStreamInfo = new EventStreamInfo();
        // Call the selectObjectContent method with the request and a response
handler.
        // Supply an EventStreamInfo object to the response handler to gather
records and information from the response.
        s3AsyncClient.selectObjectContent(select,
buildResponseHandler(eventStreamInfo)).join();
```

```
        // Log out information gathered while processing the response stream.
        long recordCount = eventStreamInfo.getRecords().stream().mapToInt(record
 ->
                record.split("\n").length
        ).sum();
        logger.info("Total records {}: {}", fileType, recordCount);
        logger.info("Visitor onRecords for fileType {} called {} times",
 fileType, eventStreamInfo.getCountOnRecordsCalled());
        logger.info("Visitor onStats for fileType {}, {}", fileType,
 eventStreamInfo.getStats());
        logger.info("Visitor onContinuations for fileType {}, {}", fileType,
 eventStreamInfo.getCountContinuationEvents());
        return eventStreamInfo;
    }

    static SelectObjectContentResponseHandler
 buildResponseHandler(EventStreamInfo eventStreamInfo) {
        // Use a Visitor to process the response stream. This visitor logs
 information and gathers details while processing.
        final SelectObjectContentResponseHandler.Visitor visitor =
 SelectObjectContentResponseHandler.Visitor.builder()
                .onRecords(r -> {
                    logger.info("Record event received.");
                    eventStreamInfo.addRecord(r.payload().asUtf8String());
                    eventStreamInfo.incrementOnRecordsCalled();
                })
                .onCont(ce -> {
                    logger.info("Continuation event received.");
                    eventStreamInfo.incrementContinuationEvents();
                })
                .onProgress(pe -> {
                    Progress progress = pe.details();
                    logger.info("Progress event received:\n bytesScanned:
{}\nbytesProcessed: {}\nbytesReturned:{}",
                            progress.bytesScanned(),
                            progress.bytesProcessed(),
                            progress.bytesReturned());
                })
                .onEnd(ee -> logger.info("End event received."))
                .onStats(se -> {
                    logger.info("Stats event received.");
                    eventStreamInfo.addStats(se.details());
                })
```

```
                .build();

        // Build the SelectObjectContentResponseHandler with the visitor that
processes the stream.
        return SelectObjectContentResponseHandler.builder()
                .subscriber(visitor).build();
    }

    // The EventStreamInfo class is used to store information gathered while
processing the response stream.
    static class EventStreamInfo {
        private final List<String> records = new ArrayList<>();
        private Integer countOnRecordsCalled = 0;
        private Integer countContinuationEvents = 0;
        private Stats stats;

        void incrementOnRecordsCalled() {
            countOnRecordsCalled++;
        }

        void incrementContinuationEvents() {
            countContinuationEvents++;
        }

        void addRecord(String record) {
            records.add(record);
        }

        void addStats(Stats stats) {
            this.stats = stats;
        }

        public List<String> getRecords() {
            return records;
        }

        public Integer getCountOnRecordsCalled() {
            return countOnRecordsCalled;
        }

        public Integer getCountContinuationEvents() {
            return countContinuationEvents;
        }
```

```
        public Stats getStats() {
            return stats;
        }
    }
```

- Untuk detail API, lihat SelectObjectContentdi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`UploadPart`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`UploadPart`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Melakukan pengunggahan multibagian
- Gunakan checksum

CLI

AWS CLI

Perintah berikut mengunggah bagian pertama dalam unggahan multibagian yang dimulai dengan perintah: `create-multipart-upload`

```
aws s3api upload-part --bucket my-bucket --key 'multipart/01' --part-number 1 --
body part01 --upload-id
 "dfRtDYU0WWCCcH43C3WFbkRONycyCpTJJvxu2i5GYkZljF.Yxwh6XG7WfS2vC4to6HiV6Yjlx.cph0gtNBtJ8P3
```

bodyOpsi ini mengambil nama atau jalur file lokal untuk diunggah (jangan gunakan awalan file://). Ukuran bagian minimum adalah 5 MB. Upload ID dikembalikan oleh `create-multipart-upload` dan juga dapat diambil dengan`list-multipart-uploads`. Bucket dan kunci ditentukan saat Anda membuat unggahan multipart.

Output:

```
{
    "ETag": "\"e868e0f4719e394144ef36531ee6824c\""
}
```

Simpan nilai ETag dari setiap bagian untuk nanti. Mereka diminta untuk menyelesaikan unggahan multipart.

- Untuk detail API, lihat UploadPartdi Referensi AWS CLI Perintah.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        let upload_part_res = client
            .upload_part()
            .key(&key)
            .bucket(&bucket_name)
            .upload_id(upload_id)
            .body(stream)
            .part_number(part_number)
            .send()
            .await?;
        upload_parts.push(
            CompletedPart::builder()
                .e_tag(upload_part_res.e_tag.unwrap_or_default())
                .part_number(part_number)
                .build(),
        );

    let completed_multipart_upload: CompletedMultipartUpload =
CompletedMultipartUpload::builder()
        .set_parts(Some(upload_parts))
        .build();
```

- Untuk detail API, lihat UploadPartreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Skenario untuk Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara menerapkan skenario umum di Amazon S3 dengan AWS SDK. Skenario ini menunjukkan cara menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam Amazon S3. Setiap skenario menyertakan tautan ke GitHub, di mana Anda dapat menemukan petunjuk tentang cara mengatur dan menjalankan kode.

Contoh

- Membuat URL yang telah ditetapkan sebelumnya untuk Amazon S3 menggunakan SDK AWS
- Halaman web yang mencantumkan objek Amazon S3 menggunakan SDK AWS
- Hapus unggahan multipart yang tidak lengkap ke Amazon S3 menggunakan SDK AWS
- Mengunduh semua objek di bucket Amazon Simple Storage Service (Amazon S3) ke direktori lokal
- Mendapatkan objek Amazon S3 dari Titik Akses Multi-Wilayah dengan menggunakan SDK AWS
- Dapatkan objek dari bucket Amazon S3 menggunakan AWS SDK, dengan menentukan header If-Modified-Since
- Memulai bucket dan objek Amazon S3 menggunakan SDK AWS
- Memulai enkripsi untuk objek Amazon S3 menggunakan SDK AWS
- Memulai tag untuk objek Amazon S3 menggunakan SDK AWS
- Dapatkan konfigurasi penahanan hukum objek Amazon S3 menggunakan SDK AWS
- Bekerja dengan fitur kunci objek Amazon S3 menggunakan SDK AWS
- Mengelola daftar kontrol akses (ACL) untuk bucket Amazon S3 menggunakan SDK AWS
- Mengelola objek Amazon S3 berversi dalam batch dengan fungsi Lambda menggunakan SDK AWS
- Mengurai URI Amazon S3 menggunakan SDK AWS
- Melakukan salinan multi-bagian objek Amazon S3 menggunakan SDK AWS
- Melakukan pengunggahan multi-bagian objek Amazon S3 menggunakan SDK AWS
- Lacak unggahan atau unduh objek Amazon S3 menggunakan SDK AWS

- Contoh pendekatan untuk pengujian unit dan integrasi dengan AWS SDK
- Mengunggah direktori lokal secara rekursif ke bucket Amazon Simple Storage Service (Amazon S3)
- Unggah atau unduh file besar ke dan dari Amazon S3 menggunakan SDK AWS
- Unggah aliran dengan ukuran yang tidak diketahui ke objek Amazon S3 menggunakan SDK AWS
- Menggunakan checksum untuk bekerja dengan objek Amazon S3 menggunakan SDK AWS
- Bekerja dengan objek berversi Amazon S3 menggunakan SDK AWS

Membuat URL yang telah ditetapkan sebelumnya untuk Amazon S3 menggunakan SDK AWS

Contoh kode berikut ini menunjukkan cara membuat URL yang telah ditandatangani untuk Amazon S3 dan mengunggah objek.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat URL yang telah ditetapkan sebelumnya yang dapat melakukan tindakan Amazon S3 untuk waktu yang terbatas.

```
using System;
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;

public class GenPresignedUrl
{
    public static void Main()
    {
        const string bucketName = "doc-example-bucket";
        const string objectKey = "sample.txt";
```

```
            // Specify how long the presigned URL lasts, in hours
            const double timeoutDuration = 12;

            // Specify the AWS Region of your Amazon S3 bucket. If it is
            // different from the Region defined for the default user,
            // pass the Region to the constructor for the client. For
            // example: new AmazonS3Client(RegionEndpoint.USEast1);

            // If using the Region us-east-1, and server-side encryption with AWS
 KMS, you must specify Signature Version 4.
            // Region us-east-1 defaults to Signature Version 2 unless explicitly
 set to Version 4 as shown below.
            // For more details, see https://docs.aws.amazon.com/AmazonS3/latest/
userguide/UsingAWSSDK.html#specify-signature-version
            // and https://docs.aws.amazon.com/sdkfornet/v3/apidocs/items/Amazon/
TAWSConfigsS3.html
            AWSConfigsS3.UseSignatureVersion4 = true;
            IAmazonS3 s3Client = new AmazonS3Client(RegionEndpoint.USEast1);

            string urlString = GeneratePresignedURL(s3Client, bucketName,
 objectKey, timeoutDuration);
            Console.WriteLine($"The generated URL is: {urlString}.");
        }

        /// <summary>
        /// Generate a presigned URL that can be used to access the file named
        /// in the objectKey parameter for the amount of time specified in the
        /// duration parameter.
        /// </summary>
        /// <param name="client">An initialized S3 client object used to call
        /// the GetPresignedUrl method.</param>
        /// <param name="bucketName">The name of the S3 bucket containing the
        /// object for which to create the presigned URL.</param>
        /// <param name="objectKey">The name of the object to access with the
        /// presigned URL.</param>
        /// <param name="duration">The length of time for which the presigned
        /// URL will be valid.</param>
        /// <returns>A string representing the generated presigned URL.</returns>
        public static string GeneratePresignedURL(IAmazonS3 client, string
 bucketName, string objectKey, double duration)
        {
            string urlString = string.Empty;
            try
```

```
            {
                var request = new GetPreSignedUrlRequest()
                {
                    BucketName = bucketName,
                    Key = objectKey,
                    Expires = DateTime.UtcNow.AddHours(duration),
                };
                urlString = client.GetPreSignedURL(request);
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error:'{ex.Message}'");
            }

            return urlString;
        }
    }
```

Buat URL yang telah ditetapkan sebelumnya dan unggah menggunakan URL tersebut.

```
    using System;
    using System.IO;
    using System.Net.Http;
    using System.Threading.Tasks;
    using Amazon;
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example shows how to upload an object to an Amazon Simple Storage
    /// Service (Amazon S3) bucket using a presigned URL. The code first
    /// creates a presigned URL and then uses it to upload an object to an
    /// Amazon S3 bucket using that URL.
    /// </summary>
    public class UploadUsingPresignedURL
    {
        private static HttpClient httpClient = new HttpClient();

        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";
```

```
            string keyName = "samplefile.txt";
            string filePath = $"source\\{keyName}";

            // Specify how long the signed URL will be valid in hours.
            double timeoutDuration = 12;

            // Specify the AWS Region of your Amazon S3 bucket. If it is
            // different from the Region defined for the default user,
            // pass the Region to the constructor for the client. For
            // example: new AmazonS3Client(RegionEndpoint.USEast1);

            // If using the Region us-east-1, and server-side encryption with AWS
 KMS, you must specify Signature Version 4.
            // Region us-east-1 defaults to Signature Version 2 unless explicitly
 set to Version 4 as shown below.
            // For more details, see https://docs.aws.amazon.com/AmazonS3/latest/
userguide/UsingAWSSDK.html#specify-signature-version
            // and https://docs.aws.amazon.com/sdkfornet/v3/apidocs/items/Amazon/
TAWSConfigsS3.html
            AWSConfigsS3.UseSignatureVersion4 = true;
            IAmazonS3 client = new AmazonS3Client(RegionEndpoint.USEast1);

            var url = GeneratePreSignedURL(client, bucketName, keyName,
 timeoutDuration);
            var success = await UploadObject(filePath, url);

            if (success)
            {
                Console.WriteLine("Upload succeeded.");
            }
            else
            {
                Console.WriteLine("Upload failed.");
            }
        }

        /// <summary>
        /// Uploads an object to an Amazon S3 bucket using the presigned URL
 passed in
        /// the url parameter.
        /// </summary>
        /// <param name="filePath">The path (including file name) to the local
        /// file you want to upload.</param>
        /// <param name="url">The presigned URL that will be used to upload the
```

```
        /// file to the Amazon S3 bucket.</param>
        /// <returns>A Boolean value indicating the success or failure of the
        /// operation, based on the HttpWebResponse.</returns>
        public static async Task<bool> UploadObject(string filePath, string url)
        {
            using var streamContent = new StreamContent(
                new FileStream(filePath, FileMode.Open, FileAccess.Read));

            var response = await httpClient.PutAsync(url, streamContent);
            return response.IsSuccessStatusCode;
        }

        /// <summary>
        /// Generates a presigned URL which will be used to upload an object to
        /// an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
 call
        /// GetPreSignedURL.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket to which
 the
        /// presigned URL will point.</param>
        /// <param name="objectKey">The name of the file that will be uploaded.</
param>
        /// <param name="duration">How long (in hours) the presigned URL will
        /// be valid.</param>
        /// <returns>The generated URL.</returns>
        public static string GeneratePreSignedURL(
            IAmazonS3 client,
            string bucketName,
            string objectKey,
            double duration)
        {
            var request = new GetPreSignedUrlRequest
            {
                BucketName = bucketName,
                Key = objectKey,
                Verb = HttpVerb.PUT,
                Expires = DateTime.UtcNow.AddHours(duration),
            };

            string url = client.GetPreSignedURL(request);
            return url;
        }
```

```
    }
```

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Hasilkan URL yang telah ditandatangani sebelumnya untuk mengunduh objek.

```
//! Routine which demonstrates creating a pre-signed URL to download an object
 from an
//! Amazon Simple Storage Service (Amazon S3) bucket.
/*!
  \param bucketName: Name of the bucket.
  \param key: Name of an object key.
  \param expirationSeconds: Expiration in seconds for pre-signed URL.
  \param clientConfig: Aws client configuration.
  \return Aws::String: A pre-signed URL.
*/
Aws::String AwsDoc::S3::generatePreSignedGetObjectUrl(const Aws::String
 &bucketName,
                                                       const Aws::String &key,
                                                       uint64_t expirationSeconds,
                                                       const
 Aws::S3::S3ClientConfiguration &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    return client.GeneratePresignedUrl(bucketName, key,
 Aws::Http::HttpMethod::HTTP_GET,
                                        expirationSeconds);
}
```

Unduh menggunakan libcurl.

```
static size_t myCurlWriteBack(char *buffer, size_t size, size_t nitems, void
 *userdata) {
    Aws::StringStream *str = (Aws::StringStream *) userdata;

    if (nitems > 0) {
        str->write(buffer, size * nitems);
    }
    return size * nitems;
}

//! Utility routine to test getObject with a pre-signed URL.
/*!
  \param presignedURL: A pre-signed URL to get an object from a bucket.
  \param resultString: A string to hold the result.
  \return bool: Function succeeded.
*/
bool AwsDoc::S3::getObjectWithPresignedObjectUrl(const Aws::String &presignedURL,
                                                 Aws::String &resultString) {
    CURL *curl = curl_easy_init();
    CURLcode result;

    std::stringstream outWriteString;

    result = curl_easy_setopt(curl, CURLOPT_WRITEDATA, &outWriteString);

    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_WRITEDATA " << std::endl;
        return false;
    }

    result = curl_easy_setopt(curl, CURLOPT_WRITEFUNCTION, myCurlWriteBack);

    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_WRITEFUNCTION" << std::endl;
        return false;
    }

    result = curl_easy_setopt(curl, CURLOPT_URL, presignedURL.c_str());

    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_URL" << std::endl;
        return false;
    }
```

```
    result = curl_easy_perform(curl);

    if (result != CURLE_OK) {
        std::cerr << "Failed to perform CURL request" << std::endl;
        return false;
    }

    resultString = outWriteString.str();

    if (resultString.find("<?xml") == 0) {
        std::cerr << "Failed to get object, response:\n" << resultString <<
 std::endl;
        return false;
    }

    return true;
}
```

Buat URL yang telah ditandatangani sebelumnya untuk mengunggah objek.

```
//! Routine which demonstrates creating a pre-signed URL to upload an object to
 an
//! Amazon Simple Storage Service (Amazon S3) bucket.
/*!
  \param bucketName: Name of the bucket.
  \param key: Name of an object key.
  \param clientConfig: Aws client configuration.
  \return Aws::String: A pre-signed URL.
*/
Aws::String AwsDoc::S3::generatePreSignedPutObjectUrl(const Aws::String
 &bucketName,
                                                       const Aws::String &key,
                                                       uint64_t expirationSeconds,
                                                       const
 Aws::S3::S3ClientConfiguration &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    return client.GeneratePresignedUrl(bucketName, key,
 Aws::Http::HttpMethod::HTTP_PUT,
                                        expirationSeconds);
}
```

Unggah menggunakan libcurl.

```
static size_t myCurlReadBack(char *buffer, size_t size, size_t nitems, void
 *userdata) {
    Aws::StringStream *str = (Aws::StringStream *) userdata;

    str->read(buffer, size * nitems);

    return str->gcount();
}

static size_t myCurlWriteBack(char *buffer, size_t size, size_t nitems, void
 *userdata) {
    Aws::StringStream *str = (Aws::StringStream *) userdata;

    if (nitems > 0) {
        str->write(buffer, size * nitems);
    }
    return size * nitems;
}

//! Utility routine to test putObject with a pre-signed URL.
/*!
  \param presignedURL: A pre-signed URL to put an object in a bucket.
  \param data: Body of the putObject request.
  \return bool: Function succeeded.
*/
bool AwsDoc::S3::PutStringWithPresignedObjectURL(const Aws::String &presignedURL,
                                                 const Aws::String &data) {
    CURL *curl = curl_easy_init();
    CURLcode result;

    Aws::StringStream readStringStream;
    readStringStream << data;
    result = curl_easy_setopt(curl, CURLOPT_READFUNCTION, myCurlReadBack);

    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_READFUNCTION" << std::endl;
        return false;
    }

    result = curl_easy_setopt(curl, CURLOPT_READDATA, &readStringStream);
    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_READDATA" << std::endl;
```

```
        return false;
    }

    result = curl_easy_setopt(curl, CURLOPT_INFILESIZE_LARGE,
                              (curl_off_t) data.size());

    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_INFILESIZE_LARGE" << std::endl;
        return false;
    }

    result = curl_easy_setopt(curl, CURLOPT_WRITEFUNCTION, myCurlWriteBack);

    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_WRITEFUNCTION" << std::endl;
        return false;
    }

    std::stringstream outWriteString;

    result = curl_easy_setopt(curl, CURLOPT_WRITEDATA, &outWriteString);

    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_WRITEDATA " << std::endl;
        return false;
    }

    result = curl_easy_setopt(curl, CURLOPT_URL, presignedURL.c_str());

    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_URL" << std::endl;
        return false;
    }

    result = curl_easy_setopt(curl, CURLOPT_UPLOAD, 1L);

    if (result != CURLE_OK) {
        std::cerr << "Failed to set CURLOPT_PUT" << std::endl;
        return false;
    }

    result = curl_easy_perform(curl);

    if (result != CURLE_OK) {
```

```
        std::cerr << "Failed to perform CURL request" << std::endl;
        return false;
    }

    std::string outString = outWriteString.str();
    if (outString.empty()) {
        std::cout << "Successfully put object." << std::endl;
        return true;
    } else {
        std::cout << "A server error was encountered, output:\n" << outString
                  << std::endl;
        return false;
    }
}
```

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat fungsi yang membungkus tindakan S3 yang telah ditetapkan sebelumnya.

```
// Presigner encapsulates the Amazon Simple Storage Service (Amazon S3) presign
 actions
// used in the examples.
// It contains PresignClient, a client that is used to presign requests to Amazon
 S3.
// Presigned requests contain temporary credentials and can be made from any HTTP
 client.
type Presigner struct {
 PresignClient *s3.PresignClient
}
```

```
// GetObject makes a presigned request that can be used to get an object from a
 bucket.
// The presigned request is valid for the specified number of seconds.
func (presigner Presigner) GetObject(
 bucketName string, objectKey string, lifetimeSecs int64)
 (*v4.PresignedHTTPRequest, error) {
 request, err := presigner.PresignClient.PresignGetObject(context.TODO(),
 &s3.GetObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
 }, func(opts *s3.PresignOptions) {
  opts.Expires = time.Duration(lifetimeSecs * int64(time.Second))
 })
 if err != nil {
  log.Printf("Couldn't get a presigned request to get %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }
 return request, err
}



// PutObject makes a presigned request that can be used to put an object in a
 bucket.
// The presigned request is valid for the specified number of seconds.
func (presigner Presigner) PutObject(
 bucketName string, objectKey string, lifetimeSecs int64)
 (*v4.PresignedHTTPRequest, error) {
 request, err := presigner.PresignClient.PresignPutObject(context.TODO(),
 &s3.PutObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
 }, func(opts *s3.PresignOptions) {
  opts.Expires = time.Duration(lifetimeSecs * int64(time.Second))
 })
 if err != nil {
  log.Printf("Couldn't get a presigned request to put %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }
 return request, err
}
```

```
// DeleteObject makes a presigned request that can be used to delete an object
 from a bucket.
func (presigner Presigner) DeleteObject(bucketName string, objectKey string)
 (*v4.PresignedHTTPRequest, error) {
 request, err := presigner.PresignClient.PresignDeleteObject(context.TODO(),
 &s3.DeleteObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't get a presigned request to delete object %v. Here's why:
 %v\n", objectKey, err)
 }
 return request, err
}
```

Jalankan contoh interaktif yang menghasilkan dan menggunakan URL yang telah ditetapkan sebelumnya untuk mengunggah, mengunduh, dan menghapus objek S3.

```
// RunPresigningScenario is an interactive example that shows you how to get
 presigned
// HTTP requests that you can use to move data into and out of Amazon Simple
 Storage
// Service (Amazon S3). The presigned requests contain temporary credentials and
 can
// be used by an HTTP client.
//
// 1. Get a presigned request to put an object in a bucket.
// 2. Use the net/http package to use the presigned request to upload a local
 file to the bucket.
// 3. Get a presigned request to get an object from a bucket.
// 4. Use the net/http package to use the presigned request to download the
 object to a local file.
// 5. Get a presigned request to delete an object from a bucket.
// 6. Use the net/http package to use the presigned request to delete the object.
//
// This example creates an Amazon S3 presign client from the specified sdkConfig
 so that
// you can replace it with a mocked or stubbed config for unit testing.
//
```

```
// It uses a questioner from the `demotools` package to get input during the
 example.
// This package can be found in the ..\..\demotools folder of this repo.
//
// It uses an IHttpRequester interface to abstract HTTP requests so they can be
 mocked
// during testing.
func RunPresigningScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner, httpRequester IHttpRequester) {
 defer func() {
  if r := recover(); r != nil {
   fmt.Printf("Something went wrong with the demo.")
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the Amazon S3 presigning demo.")
 log.Println(strings.Repeat("-", 88))

 s3Client := s3.NewFromConfig(sdkConfig)
 bucketBasics := actions.BucketBasics{S3Client: s3Client}
 presignClient := s3.NewPresignClient(s3Client)
 presigner := actions.Presigner{PresignClient: presignClient}

 bucketName := questioner.Ask("We'll need a bucket. Enter a name for a bucket "+
  "you own or one you want to create:", demotools.NotEmpty{})
 bucketExists, err := bucketBasics.BucketExists(bucketName)
 if err != nil {
  panic(err)
 }
 if !bucketExists {
  err = bucketBasics.CreateBucket(bucketName, sdkConfig.Region)
  if err != nil {
   panic(err)
  } else {
   log.Println("Bucket created.")
  }
 }
 log.Println(strings.Repeat("-", 88))

 log.Printf("Let's presign a request to upload a file to your bucket.")
 uploadFilename := questioner.Ask("Enter the path to a file you want to upload:",
  demotools.NotEmpty{})
 uploadKey := questioner.Ask("What would you like to name the uploaded object?",
```

```
 demotools.NotEmpty{})
uploadFile, err := os.Open(uploadFilename)
if err != nil {
 panic(err)
}
defer uploadFile.Close()
presignedPutRequest, err := presigner.PutObject(bucketName, uploadKey, 60)
if err != nil {
 panic(err)
}
log.Printf("Got a presigned %v request to URL:\n\t%v\n",
presignedPutRequest.Method,
 presignedPutRequest.URL)
log.Println("Using net/http to send the request...")
info, err := uploadFile.Stat()
if err != nil {
 panic(err)
}
putResponse, err := httpRequester.Put(presignedPutRequest.URL, info.Size(),
uploadFile)
if err != nil {
 panic(err)
}
log.Printf("%v object %v with presigned URL returned %v.",
presignedPutRequest.Method,
 uploadKey, putResponse.StatusCode)
log.Println(strings.Repeat("-", 88))

log.Printf("Let's presign a request to download the object.")
questioner.Ask("Press Enter when you're ready.")
presignedGetRequest, err := presigner.GetObject(bucketName, uploadKey, 60)
if err != nil {
 panic(err)
}
log.Printf("Got a presigned %v request to URL:\n\t%v\n",
presignedGetRequest.Method,
 presignedGetRequest.URL)
log.Println("Using net/http to send the request...")
getResponse, err := httpRequester.Get(presignedGetRequest.URL)
if err != nil {
 panic(err)
}
log.Printf("%v object %v with presigned URL returned %v.",
presignedGetRequest.Method,
```

```
  uploadKey, getResponse.StatusCode)
 defer getResponse.Body.Close()
 downloadBody, err := io.ReadAll(getResponse.Body)
 if err != nil {
  panic(err)
 }
 log.Printf("Downloaded %v bytes. Here are the first 100 of them:\n",
 len(downloadBody))
 log.Println(strings.Repeat("-", 88))
 log.Println(string(downloadBody[:100]))
 log.Println(strings.Repeat("-", 88))

 log.Println("Let's presign a request to delete the object.")
 questioner.Ask("Press Enter when you're ready.")
 presignedDelRequest, err := presigner.DeleteObject(bucketName, uploadKey)
 if err != nil {
  panic(err)
 }
 log.Printf("Got a presigned %v request to URL:\n\t%v\n",
 presignedDelRequest.Method,
  presignedDelRequest.URL)
 log.Println("Using net/http to send the request...")
 delResponse, err := httpRequester.Delete(presignedDelRequest.URL)
 if err != nil {
  panic(err)
 }
 log.Printf("%v object %v with presigned URL returned %v.\n",
 presignedDelRequest.Method,
  uploadKey, delResponse.StatusCode)
 log.Println(strings.Repeat("-", 88))

 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Tentukan pembungkus permintaan HTTP yang digunakan oleh contoh untuk membuat permintaan HTTP.

```
// IHttpRequester abstracts HTTP requests into an interface so it can be mocked
 during
```

```
// unit testing.
type IHttpRequester interface {
 Get(url string) (resp *http.Response, err error)
 Put(url string, contentLength int64, body io.Reader) (resp *http.Response, err
 error)
 Delete(url string) (resp *http.Response, err error)
}

// HttpRequester uses the net/http package to make HTTP requests during the
 scenario.
type HttpRequester struct{}

func (httpReq HttpRequester) Get(url string) (resp *http.Response, err error) {
 return http.Get(url)
}
func (httpReq HttpRequester) Put(url string, contentLength int64, body io.Reader)
 (resp *http.Response, err error) {
 putRequest, err := http.NewRequest("PUT", url, body)
 if err != nil {
  return nil, err
 }
 putRequest.ContentLength = contentLength
 return http.DefaultClient.Do(putRequest)
}
func (httpReq HttpRequester) Delete(url string) (resp *http.Response, err error)
 {
 delRequest, err := http.NewRequest("DELETE", url, nil)
 if err != nil {
  return nil, err
 }
 return http.DefaultClient.Do(delRequest)
}
```

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat URL yang telah ditandatangani sebelumnya untuk suatu objek, lalu unduh (GET request).

Impor.

```
import com.example.s3.util.PresignUrlUtils;
import org.slf4j.Logger;
import software.amazon.awssdk.http.HttpExecuteRequest;
import software.amazon.awssdk.http.HttpExecuteResponse;
import software.amazon.awssdk.http.SdkHttpClient;
import software.amazon.awssdk.http.SdkHttpMethod;
import software.amazon.awssdk.http.SdkHttpRequest;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.presigner.S3Presigner;
import
 software.amazon.awssdk.services.s3.presigner.model.GetObjectPresignRequest;
import
 software.amazon.awssdk.services.s3.presigner.model.PresignedGetObjectRequest;
import software.amazon.awssdk.utils.IoUtils;

import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.URISyntaxException;
import java.net.URL;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;
import java.nio.file.Paths;
```

```
import java.time.Duration;
import java.util.UUID;
```

Hasilkan URL.

```
    /* Create a pre-signed URL to download an object in a subsequent GET request.
 */
    public String createPresignedGetUrl(String bucketName, String keyName) {
        try (S3Presigner presigner = S3Presigner.create()) {

            GetObjectRequest objectRequest = GetObjectRequest.builder()
                    .bucket(bucketName)
                    .key(keyName)
                    .build();

            GetObjectPresignRequest presignRequest =
 GetObjectPresignRequest.builder()
                    .signatureDuration(Duration.ofMinutes(10))  // The URL will
 expire in 10 minutes.
                    .getObjectRequest(objectRequest)
                    .build();

            PresignedGetObjectRequest presignedRequest =
 presigner.presignGetObject(presignRequest);
            logger.info("Presigned URL: [{}]",
 presignedRequest.url().toString());
            logger.info("HTTP method: [{}]",
 presignedRequest.httpRequest().method());

            return presignedRequest.url().toExternalForm();
        }
    }
```

Unduh objek dengan menggunakan salah satu dari tiga pendekatan berikut.

Gunakan kelas JDK `HttpURLConnection` (sejak v1.1) untuk melakukan download.

```
    /* Use the JDK HttpURLConnection (since v1.1) class to do the download. */
    public byte[] useHttpUrlConnectionToGet(String presignedUrlString) {
        ByteArrayOutputStream byteArrayOutputStream = new
 ByteArrayOutputStream(); // Capture the response body to a byte array.
```

```
        try {
            URL presignedUrl = new URL(presignedUrlString);
            HttpURLConnection connection = (HttpURLConnection)
presignedUrl.openConnection();
            connection.setRequestMethod("GET");
            // Download the result of executing the request.
            try (InputStream content = connection.getInputStream()) {
                IoUtils.copy(content, byteArrayOutputStream);
            }
            logger.info("HTTP response code is " + connection.getResponseCode());

        } catch (S3Exception | IOException e) {
            logger.error(e.getMessage(), e);
        }
        return byteArrayOutputStream.toByteArray();
    }
```

Gunakan kelas JDK `HttpClient` (sejak v11) untuk melakukan download.

```
    /* Use the JDK HttpClient (since v11) class to do the download. */
    public byte[] useHttpClientToGet(String presignedUrlString) {
        ByteArrayOutputStream byteArrayOutputStream = new
ByteArrayOutputStream(); // Capture the response body to a byte array.

        HttpRequest.Builder requestBuilder = HttpRequest.newBuilder();
        HttpClient httpClient = HttpClient.newHttpClient();
        try {
            URL presignedUrl = new URL(presignedUrlString);
            HttpResponse<InputStream> response = httpClient.send(requestBuilder
                            .uri(presignedUrl.toURI())
                            .GET()
                            .build(),
                    HttpResponse.BodyHandlers.ofInputStream());

            IoUtils.copy(response.body(), byteArrayOutputStream);

            logger.info("HTTP response code is " + response.statusCode());

        } catch (URISyntaxException | InterruptedException | IOException e) {
            logger.error(e.getMessage(), e);
        }
```

```
        return byteArrayOutputStream.toByteArray();
    }
```

Gunakan kelas AWS SDK for `SdkHttpClient` Java untuk melakukan download.

```
    /* Use the AWS SDK for Java SdkHttpClient class to do the download. */
    public byte[] useSdkHttpClientToPut(String presignedUrlString) {

        ByteArrayOutputStream byteArrayOutputStream = new
ByteArrayOutputStream(); // Capture the response body to a byte array.
        try {
            URL presignedUrl = new URL(presignedUrlString);
            SdkHttpRequest request = SdkHttpRequest.builder()
                    .method(SdkHttpMethod.GET)
                    .uri(presignedUrl.toURI())
                    .build();

            HttpExecuteRequest executeRequest = HttpExecuteRequest.builder()
                    .request(request)
                    .build();

            try (SdkHttpClient sdkHttpClient = ApacheHttpClient.create()) {
                HttpExecuteResponse response =
sdkHttpClient.prepareRequest(executeRequest).call();
                response.responseBody().ifPresentOrElse(
                        abortableInputStream -> {
                            try {
                                IoUtils.copy(abortableInputStream,
byteArrayOutputStream);
                            } catch (IOException e) {
                                throw new RuntimeException(e);
                            }
                        },
                        () -> logger.error("No response body."));

                logger.info("HTTP Response code is {}",
response.httpResponse().statusCode());
            }
        } catch (URISyntaxException | IOException e) {
            logger.error(e.getMessage(), e);
        }
        return byteArrayOutputStream.toByteArray();
```

```
    }
```

Buat URL yang telah ditandatangani sebelumnya untuk unggahan, lalu unggah file (permintaan PUT).

Impor.

```
import com.example.s3.util.PresignUrlUtils;
import org.slf4j.Logger;
import software.amazon.awssdk.core.internal.sync.FileContentStreamProvider;
import software.amazon.awssdk.http.HttpExecuteRequest;
import software.amazon.awssdk.http.HttpExecuteResponse;
import software.amazon.awssdk.http.SdkHttpClient;
import software.amazon.awssdk.http.SdkHttpMethod;
import software.amazon.awssdk.http.SdkHttpRequest;
import software.amazon.awssdk.http.apache.ApacheHttpClient;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.presigner.S3Presigner;
import
 software.amazon.awssdk.services.s3.presigner.model.PresignedPutObjectRequest;
import
 software.amazon.awssdk.services.s3.presigner.model.PutObjectPresignRequest;

import java.io.File;
import java.io.IOException;
import java.io.OutputStream;
import java.io.RandomAccessFile;
import java.net.HttpURLConnection;
import java.net.URISyntaxException;
import java.net.URL;
import java.net.http.HttpClient;
import java.net.http.HttpRequest;
import java.net.http.HttpResponse;
import java.nio.ByteBuffer;
import java.nio.channels.FileChannel;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.time.Duration;
import java.util.Map;
import java.util.UUID;
```

Hasilkan URL.

```
   /* Create a presigned URL to use in a subsequent PUT request */
   public String createPresignedUrl(String bucketName, String keyName,
Map<String, String> metadata) {
       try (S3Presigner presigner = S3Presigner.create()) {

           PutObjectRequest objectRequest = PutObjectRequest.builder()
                   .bucket(bucketName)
                   .key(keyName)
                   .metadata(metadata)
                   .build();

           PutObjectPresignRequest presignRequest =
PutObjectPresignRequest.builder()
                   .signatureDuration(Duration.ofMinutes(10))  // The URL
expires in 10 minutes.
                   .putObjectRequest(objectRequest)
                   .build();


           PresignedPutObjectRequest presignedRequest =
presigner.presignPutObject(presignRequest);
           String myURL = presignedRequest.url().toString();
           logger.info("Presigned URL to upload a file to: [{}]", myURL);
           logger.info("HTTP method: [{}]",
presignedRequest.httpRequest().method());

           return presignedRequest.url().toExternalForm();
       }
   }
```

Unggah objek file dengan menggunakan salah satu dari tiga pendekatan berikut.

Gunakan kelas JDK `HttpURLConnection` (sejak v1.1) untuk melakukan upload.

```
   /* Use the JDK HttpURLConnection (since v1.1) class to do the upload. */
   public void useHttpUrlConnectionToPut(String presignedUrlString, File
fileToPut, Map<String, String> metadata) {
       logger.info("Begin [{}] upload", fileToPut.toString());
```

```
        try {
            URL presignedUrl = new URL(presignedUrlString);
            HttpURLConnection connection = (HttpURLConnection)
 presignedUrl.openConnection();
            connection.setDoOutput(true);
            metadata.forEach((k, v) -> connection.setRequestProperty("x-amz-
meta-" + k, v));
            connection.setRequestMethod("PUT");
            OutputStream out = connection.getOutputStream();

            try (RandomAccessFile file = new RandomAccessFile(fileToPut, "r");
                 FileChannel inChannel = file.getChannel()) {
                ByteBuffer buffer = ByteBuffer.allocate(8192); //Buffer size is
 8k

                while (inChannel.read(buffer) > 0) {
                    buffer.flip();
                    for (int i = 0; i < buffer.limit(); i++) {
                        out.write(buffer.get());
                    }
                    buffer.clear();
                }
            } catch (IOException e) {
                logger.error(e.getMessage(), e);
            }

            out.close();
            connection.getResponseCode();
            logger.info("HTTP response code is " + connection.getResponseCode());

        } catch (S3Exception | IOException e) {
            logger.error(e.getMessage(), e);
        }
    }
```

Gunakan kelas JDK `HttpClient` (sejak v11) untuk melakukan upload.

```
    /* Use the JDK HttpClient (since v11) class to do the upload. */
    public void useHttpClientToPut(String presignedUrlString, File fileToPut,
 Map<String, String> metadata) {
        logger.info("Begin [{}] upload", fileToPut.toString());
```

```
        HttpRequest.Builder requestBuilder = HttpRequest.newBuilder();
        metadata.forEach((k, v) -> requestBuilder.header("x-amz-meta-" + k, v));

        HttpClient httpClient = HttpClient.newHttpClient();
        try {
            final HttpResponse<Void> response = httpClient.send(requestBuilder
                            .uri(new URL(presignedUrlString).toURI())

.PUT(HttpRequest.BodyPublishers.ofFile(Path.of(fileToPut.toURI())))
                            .build(),
                    HttpResponse.BodyHandlers.discarding());

            logger.info("HTTP response code is " + response.statusCode());

        } catch (URISyntaxException | InterruptedException | IOException e) {
            logger.error(e.getMessage(), e);
        }
    }
```

Gunakan `SdkHttpClient` kelas AWS for Java V2 untuk melakukan upload.

```
    /* Use the AWS SDK for Java V2 SdkHttpClient class to do the upload. */
    public void useSdkHttpClientToPut(String presignedUrlString, File fileToPut,
Map<String, String> metadata) {
        logger.info("Begin [{}] upload", fileToPut.toString());

        try {
            URL presignedUrl = new URL(presignedUrlString);

            SdkHttpRequest.Builder requestBuilder = SdkHttpRequest.builder()
                    .method(SdkHttpMethod.PUT)
                    .uri(presignedUrl.toURI());
            // Add headers
            metadata.forEach((k, v) -> requestBuilder.putHeader("x-amz-meta-" +
k, v));
            // Finish building the request.
            SdkHttpRequest request = requestBuilder.build();

            HttpExecuteRequest executeRequest = HttpExecuteRequest.builder()
                    .request(request)
                    .contentStreamProvider(new
FileContentStreamProvider(fileToPut.toPath()))
```

```
                    .build();

            try (SdkHttpClient sdkHttpClient = ApacheHttpClient.create()) {
                HttpExecuteResponse response =
 sdkHttpClient.prepareRequest(executeRequest).call();
                logger.info("Response code: {}",
 response.httpResponse().statusCode());
            }
        } catch (URISyntaxException | IOException e) {
            logger.error(e.getMessage(), e);
        }
    }
```

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat URL yang telah ditetapkan sebelumnya untuk unggah objek ke bucket.

```
import https from "https";
import { PutObjectCommand, S3Client } from "@aws-sdk/client-s3";
import { fromIni } from "@aws-sdk/credential-providers";
import { HttpRequest } from "@smithy/protocol-http";
import {
  getSignedUrl,
  S3RequestPresigner,
} from "@aws-sdk/s3-request-presigner";
import { parseUrl } from "@smithy/url-parser";
import { formatUrl } from "@aws-sdk/util-format-url";
import { Hash } from "@smithy/hash-node";

const createPresignedUrlWithoutClient = async ({ region, bucket, key }) => {
  const url = parseUrl(`https://${bucket}.s3.${region}.amazonaws.com/${key}`);
  const presigner = new S3RequestPresigner({
    credentials: fromIni(),
```

```
    region,
    sha256: Hash.bind(null, "sha256"),
  });

  const signedUrlObject = await presigner.presign(
    new HttpRequest({ ...url, method: "PUT" }),
  );
  return formatUrl(signedUrlObject);
};

const createPresignedUrlWithClient = ({ region, bucket, key }) => {
  const client = new S3Client({ region });
  const command = new PutObjectCommand({ Bucket: bucket, Key: key });
  return getSignedUrl(client, command, { expiresIn: 3600 });
};

function put(url, data) {
  return new Promise((resolve, reject) => {
    const req = https.request(
      url,
      { method: "PUT", headers: { "Content-Length": new Blob([data]).size } },
      (res) => {
        let responseBody = "";
        res.on("data", (chunk) => {
          responseBody += chunk;
        });
        res.on("end", () => {
          resolve(responseBody);
        });
      },
    );
    req.on("error", (err) => {
      reject(err);
    });
    req.write(data);
    req.end();
  });
}

export const main = async () => {
  const REGION = "us-east-1";
  const BUCKET = "example_bucket";
  const KEY = "example_file.txt";
```

```
  // There are two ways to generate a presigned URL.
  // 1. Use createPresignedUrl without the S3 client.
  // 2. Use getSignedUrl in conjunction with the S3 client and GetObjectCommand.
  try {
    const noClientUrl = await createPresignedUrlWithoutClient({
      region: REGION,
      bucket: BUCKET,
      key: KEY,
    });

    const clientUrl = await createPresignedUrlWithClient({
      region: REGION,
      bucket: BUCKET,
      key: KEY,
    });

    // After you get the presigned URL, you can provide your own file
    // data. Refer to put() above.
    console.log("Calling PUT using presigned URL without client");
    await put(noClientUrl, "Hello World");

    console.log("Calling PUT using presigned URL with client");
    await put(clientUrl, "Hello World");

    console.log("\nDone. Check your S3 console.");
  } catch (err) {
    console.error(err);
  }
};
```

Buat URL yang telah ditetapkan sebelumnya untuk mengunduh objek dari bucket.

```
import { GetObjectCommand, S3Client } from "@aws-sdk/client-s3";
import { fromIni } from "@aws-sdk/credential-providers";
import { HttpRequest } from "@smithy/protocol-http";
import {
  getSignedUrl,
  S3RequestPresigner,
} from "@aws-sdk/s3-request-presigner";
import { parseUrl } from "@smithy/url-parser";
import { formatUrl } from "@aws-sdk/util-format-url";
import { Hash } from "@smithy/hash-node";
```

```
const createPresignedUrlWithoutClient = async ({ region, bucket, key }) => {
  const url = parseUrl(`https://${bucket}.s3.${region}.amazonaws.com/${key}`);
  const presigner = new S3RequestPresigner({
    credentials: fromIni(),
    region,
    sha256: Hash.bind(null, "sha256"),
  });

  const signedUrlObject = await presigner.presign(new HttpRequest(url));
  return formatUrl(signedUrlObject);
};

const createPresignedUrlWithClient = ({ region, bucket, key }) => {
  const client = new S3Client({ region });
  const command = new GetObjectCommand({ Bucket: bucket, Key: key });
  return getSignedUrl(client, command, { expiresIn: 3600 });
};

export const main = async () => {
  const REGION = "us-east-1";
  const BUCKET = "example_bucket";
  const KEY = "example_file.jpg";

  try {
    const noClientUrl = await createPresignedUrlWithoutClient({
      region: REGION,
      bucket: BUCKET,
      key: KEY,
    });

    const clientUrl = await createPresignedUrlWithClient({
      region: REGION,
      bucket: BUCKET,
      key: KEY,
    });

    console.log("Presigned URL without client");
    console.log(noClientUrl);
    console.log("\n");

    console.log("Presigned URL with client");
    console.log(clientUrl);
  } catch (err) {
```

```
    console.error(err);
  }
};
```

- Untuk informasi selengkapnya, lihat AWS SDK for JavaScript Panduan Developer.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat permintaan `GetObject` yang telah ditetapkan sebelumnya dan gunakan URL untuk mengunduh objek.

```
suspend fun getObjectPresigned(
    s3: S3Client,
    bucketName: String,
    keyName: String,
): String {
    // Create a GetObjectRequest.
    val unsignedRequest =
        GetObjectRequest {
            bucket = bucketName
            key = keyName
        }

    // Presign the GetObject request.
    val presignedRequest = s3.presignGetObject(unsignedRequest, 24.hours)

    // Use the URL from the presigned HttpRequest in a subsequent HTTP GET
 request to retrieve the object.
    val objectContents = URL(presignedRequest.url.toString()).readText()

    return objectContents
}
```

Buat permintaan yang `GetObject` telah ditetapkan sebelumnya dengan opsi lanjutan.

```
suspend fun getObjectPresignedMoreOptions(
    s3: S3Client,
    bucketName: String,
    keyName: String,
): HttpRequest {
    // Create a GetObjectRequest.
    val unsignedRequest =
        GetObjectRequest {
            bucket = bucketName
            key = keyName
        }

    // Presign the GetObject request.
    val presignedRequest =
        s3.presignGetObject(unsignedRequest, signer = CrtAwsSigner) {
            signingDate = Instant.now() + 12.hours // Presigned request can be
 used 12 hours from now.
            algorithm = AwsSigningAlgorithm.SIGV4_ASYMMETRIC
            signatureType = AwsSignatureType.HTTP_REQUEST_VIA_QUERY_PARAMS
            expiresAfter = 8.hours // Presigned request expires 8 hours later.
        }
    return presignedRequest
}
```

Buat permintaan `PutObject` yang telah ditetapkan sebelumnya dan gunakan untuk mengunggah objek.

```
suspend fun putObjectPresigned(
    s3: S3Client,
    bucketName: String,
    keyName: String,
    content: String,
) {
    // Create a PutObjectRequest.
    val unsignedRequest =
        PutObjectRequest {
            bucket = bucketName
            key = keyName
```

```
        }

    // Presign the request.
    val presignedRequest = s3.presignPutObject(unsignedRequest, 24.hours)

    // Use the URL and any headers from the presigned HttpRequest in a subsequent
 HTTP PUT request to retrieve the object.
    // Create a PUT request using the OKHttpClient API.
    val putRequest =
        Request
            .Builder()
            .url(presignedRequest.url.toString())
            .apply {
                presignedRequest.headers.forEach { key, values ->
                    header(key, values.joinToString(", "))
                }
            }.put(content.toRequestBody())
            .build()

    val response = OkHttpClient().newCall(putRequest).execute()
    assert(response.isSuccessful)
}
```

- Untuk informasi selengkapnya, lihat AWS panduan pengembang SDK untuk Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
namespace S3;
use Aws\Exception\AwsException;
use AwsUtilities\PrintableLineBreak;
use AwsUtilities\TestableReadline;
use DateTime;
```

```
require 'vendor/autoload.php';

class PresignedURL
{
    use PrintableLineBreak;
    use TestableReadline;

    public function run()
    {
        $s3Service = new S3Service();

        $expiration = new DateTime("+20 minutes");
        $linebreak = $this->getLineBreak();

        echo $linebreak;
        echo ("Welcome to the Amazon S3 presigned URL demo.\n");
        echo $linebreak;

        $bucket = $this->testable_readline("First, please enter the name of the
 S3 bucket to use: ");
        $key = $this->testable_readline("Next, provide the key of an object in
 the given bucket: ");
        echo $linebreak;
        $command = $s3Service->getClient()->getCommand('GetObject', [
            'Bucket' => $bucket,
            'Key' => $key,
        ]);
        try {
            $preSignedUrl = $s3Service->preSignedUrl($command, $expiration);
            echo "Your preSignedUrl is \n$preSignedUrl\nand will be good for the
 next 20 minutes.\n";
            echo $linebreak;
            echo "Thanks for trying the Amazon S3 presigned URL demo.\n";
        } catch (AwsException $exception) {
            echo $linebreak;
            echo "Something went wrong: $exception";
            die();
        }
    }
}

$runner = new PresignedURL();
$runner->run();
```

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat URL yang telah ditetapkan sebelumnya yang dapat melakukan tindakan S3 untuk waktu yang terbatas. Gunakan paket Permintaan untuk membuat permintaan dengan URL.

```
import argparse
import logging
import boto3
from botocore.exceptions import ClientError
import requests

logger = logging.getLogger(__name__)


def generate_presigned_url(s3_client, client_method, method_parameters,
 expires_in):
    """
    Generate a presigned Amazon S3 URL that can be used to perform an action.

    :param s3_client: A Boto3 Amazon S3 client.
    :param client_method: The name of the client method that the URL performs.
    :param method_parameters: The parameters of the specified client method.
    :param expires_in: The number of seconds the presigned URL is valid for.
    :return: The presigned URL.
    """
    try:
        url = s3_client.generate_presigned_url(
            ClientMethod=client_method, Params=method_parameters,
 ExpiresIn=expires_in
        )
        logger.info("Got presigned URL: %s", url)
```

```
    except ClientError:
        logger.exception(
            "Couldn't get a presigned URL for client method '%s'.", client_method
        )
        raise
    return url


def usage_demo():
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Welcome to the Amazon S3 presigned URL demo.")
    print("-" * 88)

    parser = argparse.ArgumentParser()
    parser.add_argument("bucket", help="The name of the bucket.")
    parser.add_argument(
        "key",
        help="For a GET operation, the key of the object in Amazon S3. For a "
        "PUT operation, the name of a file to upload.",
    )
    parser.add_argument("action", choices=("get", "put"), help="The action to
 perform.")
    args = parser.parse_args()

    s3_client = boto3.client("s3")
    client_action = "get_object" if args.action == "get" else "put_object"
    url = generate_presigned_url(
        s3_client, client_action, {"Bucket": args.bucket, "Key": args.key}, 1000
    )

    print("Using the Requests package to send a request to the URL.")
    response = None
    if args.action == "get":
        response = requests.get(url)
    elif args.action == "put":
        print("Putting data to the URL.")
        try:
            with open(args.key, "r") as object_file:
                object_text = object_file.read()
            response = requests.put(url, data=object_text)
        except FileNotFoundError:
            print(
```

```
                f"Couldn't find {args.key}. For a PUT operation, the key must be
 the "
                f"name of a file that exists on your computer."
            )

    if response is not None:
        print("Got response:")
        print(f"Status: {response.status_code}")
        print(response.text)

    print("-" * 88)


if __name__ == "__main__":
    usage_demo()
```

Buat permintaan POST yang telah ditetapkan sebelumnya untuk mengunggah file.

```
class BucketWrapper:
    """Encapsulates S3 bucket actions."""

    def __init__(self, bucket):
        """
        :param bucket: A Boto3 Bucket resource. This is a high-level resource in
 Boto3
                       that wraps bucket actions in a class-like structure.
        """
        self.bucket = bucket
        self.name = bucket.name


    def generate_presigned_post(self, object_key, expires_in):
        """
        Generate a presigned Amazon S3 POST request to upload a file.
        A presigned POST can be used for a limited time to let someone without an
 AWS
        account upload a file to a bucket.

        :param object_key: The object key to identify the uploaded object.
        :param expires_in: The number of seconds the presigned POST is valid.
        :return: A dictionary that contains the URL and form fields that contain
                 required access data.
```

```
        """
        try:
            response = self.bucket.meta.client.generate_presigned_post(
                Bucket=self.bucket.name, Key=object_key, ExpiresIn=expires_in
            )
            logger.info("Got presigned POST URL: %s", response["url"])
        except ClientError:
            logger.exception(
                "Couldn't get a presigned POST URL for bucket '%s' and object
 '%s'",
                self.bucket.name,
                object_key,
            )
            raise
        return response
```

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
require "aws-sdk-s3"
require "net/http"

# Creates a presigned URL that can be used to upload content to an object.
#
# @param bucket [Aws::S3::Bucket] An existing Amazon S3 bucket.
# @param object_key [String] The key to give the uploaded object.
# @return [URI, nil] The parsed URI if successful; otherwise nil.
def get_presigned_url(bucket, object_key)
  url = bucket.object(object_key).presigned_url(:put)
  puts "Created presigned URL: #{url}"
  URI(url)
```

```
rescue Aws::Errors::ServiceError => e
  puts "Couldn't create presigned URL for #{bucket.name}:#{object_key}. Here's
 why: #{e.message}"
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-file.txt"
  object_content = "This is the content of my-file.txt."

  bucket = Aws::S3::Bucket.new(bucket_name)
  presigned_url = get_presigned_url(bucket, object_key)
  return unless presigned_url

  response = Net::HTTP.start(presigned_url.host) do |http|
    http.send_request("PUT", presigned_url.request_uri, object_content,
 "content_type" => "")
  end

  case response
  when Net::HTTPSuccess
    puts "Content uploaded!"
  else
    puts response.value
  end
end

run_demo if $PROGRAM_NAME == __FILE__
```

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat permintaan yang ditetapkan sebelumnya ke objek GET dan PUT S3.

```
async fn get_object(
    client: &Client,
    bucket: &str,
    object: &str,
    expires_in: u64,
) -> Result<(), Box<dyn Error>> {
    let expires_in = Duration::from_secs(expires_in);
    let presigned_request = client
        .get_object()
        .bucket(bucket)
        .key(object)
        .presigned(PresigningConfig::expires_in(expires_in)?)
        .await?;

    println!("Object URI: {}", presigned_request.uri());

    Ok(())
}

async fn put_object(
    client: &Client,
    bucket: &str,
    object: &str,
    expires_in: u64,
) -> Result<(), Box<dyn Error>> {
    let expires_in = Duration::from_secs(expires_in);

    let presigned_request = client
        .put_object()
        .bucket(bucket)
        .key(object)
        .presigned(PresigningConfig::expires_in(expires_in)?)
        .await?;

    println!("Object URI: {}", presigned_request.uri());

    Ok(())
}
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Halaman web yang mencantumkan objek Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara mendaftar objek Amazon S3 di halaman web.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Kode berikut adalah komponen React yang relevan yang membuat panggilan ke AWS SDK. Versi runnable dari aplikasi yang berisi komponen ini dapat ditemukan di link sebelumnya GitHub .

```
import { useEffect, useState } from "react";
import {
  ListObjectsCommand,
  ListObjectsCommandOutput,
  S3Client,
} from "@aws-sdk/client-s3";
import { fromCognitoIdentityPool } from "@aws-sdk/credential-providers";
import "./App.css";

function App() {
  const [objects, setObjects] = useState<
    Required<ListObjectsCommandOutput>["Contents"]
  >([]);

  useEffect(() => {
    const client = new S3Client({
      region: "us-east-1",
      // Unless you have a public bucket, you'll need access to a private bucket.
```

```
      // One way to do this is to create an Amazon Cognito identity pool, attach
 a role to the pool,
      // and grant the role access to the 's3:GetObject' action.
      //
      // You'll also need to configure the CORS settings on the bucket to allow
 traffic from
      // this example site. Here's an example configuration that allows all
 origins. Don't
      // do this in production.
      //[
      //  {
      //    "AllowedHeaders": ["*"],
      //    "AllowedMethods": ["GET"],
      //    "AllowedOrigins": ["*"],
      //    "ExposeHeaders": [],
      //  },
      //]
      //
      credentials: fromCognitoIdentityPool({
        clientConfig: { region: "us-east-1" },
        identityPoolId: "<YOUR_IDENTITY_POOL_ID>",
      }),
    });
    const command = new ListObjectsCommand({ Bucket: "bucket-name" });
    client.send(command).then(({ Contents }) => setObjects(Contents || []));
  }, []);

  return (
    <div className="App">
      {objects.map((o) => (
        <div key={o.ETag}>{o.Key}</div>
      ))}
    </div>
  );
}

export default App;
```

- Untuk detail API, lihat ListObjectsdi Referensi AWS SDK for JavaScript API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Hapus unggahan multipart yang tidak lengkap ke Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara menghapus atau menghentikan unggahan multipart Amazon S3 yang tidak lengkap.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Untuk menghentikan unggahan multibagian yang sedang berlangsung atau tidak lengkap karena alasan apa pun, Anda bisa mendapatkan unggahan daftar lalu menghapusnya seperti yang ditunjukkan pada contoh berikut.

```
    public static void abortIncompleteMultipartUploadsFromList() {
        ListMultipartUploadsRequest listMultipartUploadsRequest =
ListMultipartUploadsRequest.builder()
                .bucket(bucketName)
                .build();

        ListMultipartUploadsResponse response =
s3Client.listMultipartUploads(listMultipartUploadsRequest);
        List<MultipartUpload> uploads = response.uploads();

        AbortMultipartUploadRequest abortMultipartUploadRequest;
        for (MultipartUpload upload : uploads) {
            abortMultipartUploadRequest = AbortMultipartUploadRequest.builder()
                    .bucket(bucketName)
                    .key(upload.key())
                    .expectedBucketOwner(accountId)
                    .uploadId(upload.uploadId())
```

```
                    .build();

            AbortMultipartUploadResponse abortMultipartUploadResponse =
s3Client.abortMultipartUpload(abortMultipartUploadRequest);
            if (abortMultipartUploadResponse.sdkHttpResponse().isSuccessful()) {
                logger.info("Upload ID [{}] to bucket [{}] successfully
aborted.", upload.uploadId(), bucketName);
            }
        }
    }
```

Untuk menghapus unggahan multibagian yang tidak lengkap yang dimulai sebelum atau sesudah tanggal, Anda dapat menghapus unggahan multibagian secara selektif berdasarkan titik waktu seperti yang ditunjukkan pada contoh berikut.

```
    static void abortIncompleteMultipartUploadsOlderThan(Instant pointInTime) {
        ListMultipartUploadsRequest listMultipartUploadsRequest =
ListMultipartUploadsRequest.builder()
                .bucket(bucketName)
                .build();

        ListMultipartUploadsResponse response =
s3Client.listMultipartUploads(listMultipartUploadsRequest);
        List<MultipartUpload> uploads = response.uploads();

        AbortMultipartUploadRequest abortMultipartUploadRequest;
        for (MultipartUpload upload : uploads) {
            logger.info("Found multipartUpload with upload ID [{}], initiated
[{}]", upload.uploadId(), upload.initiated());
            if (upload.initiated().isBefore(pointInTime)) {
                abortMultipartUploadRequest =
AbortMultipartUploadRequest.builder()
                        .bucket(bucketName)
                        .key(upload.key())
                        .expectedBucketOwner(accountId)
                        .uploadId(upload.uploadId())
                        .build();

                AbortMultipartUploadResponse abortMultipartUploadResponse =
s3Client.abortMultipartUpload(abortMultipartUploadRequest);
                if
(abortMultipartUploadResponse.sdkHttpResponse().isSuccessful()) {
```

```
                    logger.info("Upload ID [{}] to bucket [{}] successfully
 aborted.", upload.uploadId(), bucketName);
                }
            }
        }
    }
```

Jika Anda memiliki akses ke ID unggahan setelah memulai unggahan multibagian, Anda dapat menghapus unggahan yang sedang berlangsung dengan menggunakan ID.

```
    static void abortMultipartUploadUsingUploadId() {
        String uploadId = startUploadReturningUploadId();
        AbortMultipartUploadResponse response = s3Client.abortMultipartUpload(b -
> b
                .uploadId(uploadId)
                .bucket(bucketName)
                .key(key));

        if (response.sdkHttpResponse().isSuccessful()) {
            logger.info("Upload ID [{}] to bucket [{}] successfully aborted.",
 uploadId, bucketName);
        }
    }
```

Untuk secara konsisten menghapus unggahan multibagian yang tidak lengkap yang lebih lama dalam beberapa hari tertentu, siapkan konfigurasi siklus hidup bucket untuk bucket. Contoh berikut menunjukkan cara membuat aturan untuk menghapus unggahan yang tidak lengkap yang lebih lama dari 7 hari.

```
    static void abortMultipartUploadsUsingLifecycleConfig() {
        Collection<LifecycleRule> lifeCycleRules =
 List.of(LifecycleRule.builder()
                .abortIncompleteMultipartUpload(b -> b.
                        daysAfterInitiation(7))
                .status("Enabled")
                .filter(SdkBuilder::build) // Filter element is required.
                .build());

        // If the action is successful, the service sends back an HTTP 200
 response with an empty HTTP body.
```

```
        PutBucketLifecycleConfigurationResponse response =
s3Client.putBucketLifecycleConfiguration(b -> b
                .bucket(bucketName)
                .lifecycleConfiguration(b1 -> b1.rules(lifeCycleRules)));

        if (response.sdkHttpResponse().isSuccessful()) {
            logger.info("Rule to abort incomplete multipart uploads added to
bucket.");
        } else {
            logger.error("Unsuccessfully applied rule. HTTP status code is [{}]",
response.sdkHttpResponse().statusCode());
        }
    }
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Java 2.x .
  - AbortMultipartUpload
  - ListMultipartUploads
  - PutBucketLifecycleConfiguration

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mengunduh semua objek di bucket Amazon Simple Storage Service (Amazon S3) ke direktori lokal

Contoh Kode berikut ini menunjukkan cara mengunduh semua objek di bucket Amazon Simple Storage Service (Amazon S3) ke direktori lokal.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Gunakan S3 TransferManager untuk mengunduh semua objek S3 di bucket S3 yang sama. Lihat file lengkap dan lakukan pengujian.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.services.s3.model.ObjectIdentifier;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedDirectoryDownload;
import software.amazon.awssdk.transfer.s3.model.DirectoryDownload;
import software.amazon.awssdk.transfer.s3.model.DownloadDirectoryRequest;
import java.io.IOException;
import java.net.URI;
import java.net.URISyntaxException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.HashSet;
import java.util.Set;
import java.util.UUID;
import java.util.stream.Collectors;

    public Integer downloadObjectsToDirectory(S3TransferManager transferManager,
            URI destinationPathURI, String bucketName) {
        DirectoryDownload directoryDownload =
 transferManager.downloadDirectory(DownloadDirectoryRequest.builder()
                .destination(Paths.get(destinationPathURI))
                .bucket(bucketName)
                .build());
        CompletedDirectoryDownload completedDirectoryDownload =
 directoryDownload.completionFuture().join();

        completedDirectoryDownload.failedTransfers()
                .forEach(fail -> logger.warn("Object [{}] failed to transfer",
 fail.toString()));
        return completedDirectoryDownload.failedTransfers().size();
    }
```

- Untuk detail API, lihat DownloadDirectorydi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mendapatkan objek Amazon S3 dari Titik Akses Multi-Wilayah dengan menggunakan SDK AWS

Contoh kode berikut menunjukkan bagaimana untuk mendapatkan objek dari Multi-Region Access Point.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Konfigurasikan klien S3 untuk menggunakan algoritma penandatanganan Asymmetric Sigv4 (Sigv4a).

```
        suspend fun createS3Client(): S3Client {
            // Configure your S3Client to use the Asymmetric Sigv4 (Sigv4a)
signing algorithm.
            val sigV4AScheme = SigV4AsymmetricAuthScheme(CrtAwsSigner)
            val s3 = S3Client.fromEnvironment {
                authSchemes = listOf(sigV4AScheme)
            }
            return s3
        }
```

Gunakan ARN Titik Akses Multi-Wilayah sebagai ganti nama bucket untuk mengambil objek.

```
    suspend fun getObjectFromMrap(
        s3: S3Client,
        mrapArn: String,
        keyName: String,
```

```
    ): String? {
        val request = GetObjectRequest {
            bucket = mrapArn // Use the ARN instead of the bucket name for object
operations.
            key = keyName
        }

        var stringObj: String? = null
        s3.getObject(request) { resp ->
            stringObj = resp.body?.decodeToString()
            if (stringObj != null) {
                println("Successfully read $keyName from $mrapArn")
            }
        }
        return stringObj
    }
```

- Untuk informasi selengkapnya, lihat AWS panduan pengembang SDK untuk Kotlin.
- Untuk detail API, lihat GetObjectdi AWS SDK untuk referensi API Kotlin.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Dapatkan objek dari bucket Amazon S3 menggunakan AWS SDK, dengan menentukan header If-Modified-Since

Contoh kode berikut menunjukkan cara membaca data dari objek di bucket S3, tetapi hanya jika bucket tersebut belum dimodifikasi sejak waktu pengambilan terakhir.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
use aws_sdk_s3::{
    error::SdkError,
    operation::head_object::HeadObjectError,
    primitives::{ByteStream, DateTime, DateTimeFormat},
    Client, Error,
};
use tracing::{error, warn};

const KEY: &str = "key";
const BODY: &str = "Hello, world!";

/// Demonstrate how `if-modified-since` reports that matching objects haven't
/// changed.
///
/// # Steps
/// - Create a bucket.
/// - Put an object in the bucket.
/// - Get the bucket headers.
/// - Get the bucket headers again but only if modified.
/// - Delete the bucket.
#[tokio::main]
async fn main() -> Result<(), Error> {
    tracing_subscriber::fmt::init();

    // Get a new UUID to use when creating a unique bucket name.
    let uuid = uuid::Uuid::new_v4();

    // Load the AWS configuration from the environment.
    let client = Client::new(&aws_config::load_from_env().await);

    // Generate a unique bucket name using the previously generated UUID.
    // Then create a new bucket with that name.
    let bucket_name = format!("if-modified-since-{uuid}");
    client
        .create_bucket()
        .bucket(bucket_name.clone())
        .send()
        .await?;

    // Create a new object in the bucket whose name is `KEY` and whose
    // contents are `BODY`.
    let put_object_output = client
        .put_object()
```

```
        .bucket(bucket_name.as_str())
        .key(KEY)
        .body(ByteStream::from_static(BODY.as_bytes()))
        .send()
        .await;

    // If the `PutObject` succeeded, get the eTag string from it. Otherwise,
    // report an error and return an empty string.
    let e_tag_1 = match put_object_output {
        Ok(put_object) => put_object.e_tag.unwrap(),
        Err(err) => {
            error!("{err:?}");
            String::new()
        }
    };

    // Request the object's headers.
    let head_object_output = client
        .head_object()
        .bucket(bucket_name.as_str())
        .key(KEY)
        .send()
        .await;

    // If the `HeadObject` request succeeded, create a tuple containing the
    // values of the headers `last-modified` and `etag`. If the request
    // failed, return the error in a tuple instead.
    let (last_modified, e_tag_2) = match head_object_output {
        Ok(head_object) => (
            Ok(head_object.last_modified().cloned().unwrap()),
            head_object.e_tag.unwrap(),
        ),
        Err(err) => (Err(err), String::new()),
    };

    warn!("last modified: {last_modified:?}");
    assert_eq!(
        e_tag_1, e_tag_2,
        "PutObject and first GetObject had differing eTags"
    );

    println!("First value of last_modified: {last_modified:?}");
    println!("First tag: {}\n", e_tag_1);
```

```
    // Send a second `HeadObject` request. This time, the `if_modified_since`
    // option is specified, giving the `last_modified` value returned by the
    // first call to `HeadObject`.
    //
    // Since the object hasn't been changed, and there are no other objects in
    // the bucket, there should be no matching objects.

    let head_object_output = client
        .head_object()
        .bucket(bucket_name.as_str())
        .key(KEY)
        .if_modified_since(last_modified.unwrap())
        .send()
        .await;

    // If the `HeadObject` request succeeded, the result is a typle containing
    // the `last_modified` and `e_tag_1` properties. This is _not_ the expected
    // result.
    //
    // The _expected_ result of the second call to `HeadObject` is an
    // `SdkError::ServiceError` containing the HTTP error response. If that's
    // the case and the HTTP status is 304 (not modified), the output is a
    // tuple containing the values of the HTTP `last-modified` and `etag`
    // headers.
    //
    // If any other HTTP error occurred, the error is returned as an
    // `SdkError::ServiceError`.

    let (last_modified, e_tag_2): (Result<DateTime, SdkError<HeadObjectError>>,
String) =
        match head_object_output {
            Ok(head_object) => (
                Ok(head_object.last_modified().cloned().unwrap()),
                head_object.e_tag.unwrap(),
            ),
            Err(err) => match err {
                SdkError::ServiceError(err) => {
                    // Get the raw HTTP response. If its status is 304, the
                    // object has not changed. This is the expected code path.
                    let http = err.raw();
                    match http.status().as_u16() {
                        // If the HTTP status is 304: Not Modified, return a
                        // tuple containing the values of the HTTP
                        // `last-modified` and `etag` headers.
```

```
                        304 => (
                            Ok(DateTime::from_str(
                                http.headers().get("last-modified").unwrap(),
                                DateTimeFormat::HttpDate,
                            )
                            .unwrap()),
                            http.headers().get("etag").map(|t|
t.into()).unwrap(),
                        ),
                        // Any other HTTP status code is returned as an
                        // `SdkError::ServiceError`.
                        _ => (Err(SdkError::ServiceError(err)), String::new()),
                    }
                }
                // Any other kind of error is returned in a tuple containing the
                // error and an empty string.
                _ => (Err(err), String::new()),
            },
        };

    warn!("last modified: {last_modified:?}");
    assert_eq!(
        e_tag_1, e_tag_2,
        "PutObject and second HeadObject had different eTags"
    );

    println!("Second value of last modified: {last_modified:?}");
    println!("Second tag: {}", e_tag_2);

    // Clean up by deleting the object and the bucket.
    client
        .delete_object()
        .bucket(bucket_name.as_str())
        .key(KEY)
        .send()
        .await?;

    client
        .delete_bucket()
        .bucket(bucket_name.as_str())
        .send()
        .await?;

    Ok(())
```

```
}
```

- Untuk detail API, lihat GetObjectreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai bucket dan objek Amazon S3 menggunakan SDK AWS

Contoh kode berikut ini menunjukkan cara:

- Membuat bucket dan mengunggah file ke dalamnya.
- Mengunduh objek dari bucket.
- Menyalin objek ke subfolder di bucket.
- Membuat daftar objek dalam bucket.
- Menghapus objek bucket dan bucket tersebut.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
public class S3_Basics
{
    public static async Task Main()
    {
        // Create an Amazon S3 client object. The constructor uses the
        // default user installed on the system. To work with Amazon S3
        // features in a different AWS Region, pass the AWS Region as a
        // parameter to the client constructor.
        IAmazonS3 client = new AmazonS3Client();
        string bucketName = string.Empty;
```

```
            string filePath = string.Empty;
            string keyName = string.Empty;

            var sepBar = new string('-', Console.WindowWidth);

            Console.WriteLine(sepBar);
            Console.WriteLine("Amazon Simple Storage Service (Amazon S3) basic");
            Console.WriteLine("procedures. This application will:");
            Console.WriteLine("\n\t1. Create a bucket");
            Console.WriteLine("\n\t2. Upload an object to the new bucket");
            Console.WriteLine("\n\t3. Copy the uploaded object to a folder in the
 bucket");
            Console.WriteLine("\n\t4. List the items in the new bucket");
            Console.WriteLine("\n\t5. Delete all the items in the bucket");
            Console.WriteLine("\n\t6. Delete the bucket");
            Console.WriteLine(sepBar);

            // Create a bucket.
            Console.WriteLine($"\n{sepBar}");
            Console.WriteLine("\nCreate a new Amazon S3 bucket.\n");
            Console.WriteLine(sepBar);

            Console.Write("Please enter a name for the new bucket: ");
            bucketName = Console.ReadLine();

            var success = await S3Bucket.CreateBucketAsync(client, bucketName);
            if (success)
            {
                Console.WriteLine($"Successfully created bucket: {bucketName}.
\n");
            }
            else
            {
                Console.WriteLine($"Could not create bucket: {bucketName}.\n");
            }

            Console.WriteLine(sepBar);
            Console.WriteLine("Upload a file to the new bucket.");
            Console.WriteLine(sepBar);

            // Get the local path and filename for the file to upload.
            while (string.IsNullOrEmpty(filePath))
            {
```

```
                Console.Write("Please enter the path and filename of the file to
upload: ");
                filePath = Console.ReadLine();

                // Confirm that the file exists on the local computer.
                if (!File.Exists(filePath))
                {
                    Console.WriteLine($"Couldn't find {filePath}. Try again.\n");
                    filePath = string.Empty;
                }
            }

            // Get the file name from the full path.
            keyName = Path.GetFileName(filePath);

            success = await S3Bucket.UploadFileAsync(client, bucketName, keyName,
filePath);

            if (success)
            {
                Console.WriteLine($"Successfully uploaded {keyName} from
{filePath} to {bucketName}.\n");
            }
            else
            {
                Console.WriteLine($"Could not upload {keyName}.\n");
            }

            // Set the file path to an empty string to avoid overwriting the
            // file we just uploaded to the bucket.
            filePath = string.Empty;

            // Now get a new location where we can save the file.
            while (string.IsNullOrEmpty(filePath))
            {
                // First get the path to which the file will be downloaded.
                Console.Write("Please enter the path where the file will be
downloaded: ");
                filePath = Console.ReadLine();

                // Confirm that the file exists on the local computer.
                if (File.Exists($"{filePath}\\{keyName}"))
                {
```

```
                    Console.WriteLine($"Sorry, the file already exists in that
location.\n");
                    filePath = string.Empty;
                }
            }

            // Download an object from a bucket.
            success = await S3Bucket.DownloadObjectFromBucketAsync(client,
bucketName, keyName, filePath);

            if (success)
            {
                Console.WriteLine($"Successfully downloaded {keyName}.\n");
            }
            else
            {
                Console.WriteLine($"Sorry, could not download {keyName}.\n");
            }

            // Copy the object to a different folder in the bucket.
            string folderName = string.Empty;

            while (string.IsNullOrEmpty(folderName))
            {
                Console.Write("Please enter the name of the folder to copy your
object to: ");
                folderName = Console.ReadLine();
            }

            while (string.IsNullOrEmpty(keyName))
            {
                // Get the name to give to the object once uploaded.
                Console.Write("Enter the name of the object to copy: ");
                keyName = Console.ReadLine();
            }

            await S3Bucket.CopyObjectInBucketAsync(client, bucketName, keyName,
folderName);

            // List the objects in the bucket.
            await S3Bucket.ListBucketContentsAsync(client, bucketName);

            // Delete the contents of the bucket.
            await S3Bucket.DeleteBucketContentsAsync(client, bucketName);
```

```
            // Deleting the bucket too quickly after deleting its contents will
            // cause an error that the bucket isn't empty. So...
            Console.WriteLine("Press <Enter> when you are ready to delete the
bucket.");
            _ = Console.ReadLine();

            // Delete the bucket.
            await S3Bucket.DeleteBucketAsync(client, bucketName);
        }
    }
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for .NET .
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Bash

AWS CLI dengan skrip Bash

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
###############################################################################
# function s3_getting_started
#
# This function creates, copies, and deletes S3 buckets and objects.
#
```

```
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function s3_getting_started() {
  {
    if [ "$BUCKET_OPERATIONS_SOURCED" != "True" ]; then
      cd bucket-lifecycle-operations || exit

      source ./bucket_operations.sh
      cd ..
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the Amazon S3 getting started demo."
  echo_repeat "*" 88

  local bucket_name
  bucket_name=$(generate_random_name "doc-example-bucket")

  local region_code
  region_code=$(aws configure get region)

  if create_bucket -b "$bucket_name" -r "$region_code"; then
    echo "Created demo bucket named $bucket_name"
  else
    errecho "The bucket failed to create. This demo will exit."
    return 1
  fi

  local file_name
  while [ -z "$file_name" ]; do
    echo -n "Enter a file you want to upload to your bucket: "
    get_input
    file_name=$get_input_result

    if [ ! -f "$file_name" ]; then
      echo "Could not find file $file_name. Are you sure it exists?"
      file_name=""
    fi
  done

  local key
```

```
 key="$(basename "$file_name")"

 local result=0
 if copy_file_to_bucket "$bucket_name" "$file_name" "$key"; then
   echo "Uploaded file $file_name into bucket $bucket_name with key $key."
 else
   result=1
 fi

 local destination_file
 destination_file="$file_name.download"
 if yes_no_input "Would you like to download $key to the file $destination_file?
(y/n) "; then
   if download_object_from_bucket "$bucket_name" "$destination_file" "$key";
then
     echo "Downloaded $key in the bucket $bucket_name to the file
$destination_file."
   else
     result=1
   fi
 fi

 if yes_no_input "Would you like to copy $key a new object key in your bucket?
(y/n) "; then
   local to_key
   to_key="demo/$key"
   if copy_item_in_bucket "$bucket_name" "$key" "$to_key"; then
     echo "Copied $key in the bucket $bucket_name to the  $to_key."
   else
     result=1
   fi
 fi

 local bucket_items
 bucket_items=$(list_items_in_bucket "$bucket_name")

 # shellcheck disable=SC2181
 if [[ $? -ne 0 ]]; then
   result=1
 fi

 echo "Your bucket contains the following items."
 echo -e "Name\t\tSize"
 echo "$bucket_items"
```

```
  if yes_no_input "Delete the bucket, $bucket_name, as well as the objects in it?
 (y/n) "; then
    bucket_items=$(echo "$bucket_items" | cut -f 1)

    if delete_items_in_bucket "$bucket_name" "$bucket_items"; then
      echo "The following items were deleted from the bucket $bucket_name"
      echo "$bucket_items"
    else
      result=1
    fi

    if delete_bucket "$bucket_name"; then
      echo "Deleted the bucket $bucket_name"
    else
      result=1
    fi
  fi

  return $result
}
```

Fungsi Amazon S3 yang digunakan dalam skenario ini.

```
###############################################################################
# function create-bucket
#
# This function creates the specified bucket in the specified AWS Region, unless
# it already exists.
#
# Parameters:
#       -b bucket_name  -- The name of the bucket to create.
#       -r region_code  -- The code for an AWS Region in which to
#                          create the bucket.
#
# Returns:
#       The URL of the bucket that was created.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function create_bucket() {
```

```
  local bucket_name region_code response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function create_bucket"
    echo "Creates an Amazon S3 bucket. You must supply a bucket name:"
    echo "  -b bucket_name    The name of the bucket. It must be globally
 unique."
    echo "  [-r region_code]    The code for an AWS Region in which the bucket is
 created."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "b:r:h" option; do
    case "${option}" in
      b) bucket_name="${OPTARG}" ;;
      r) region_code="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  if [[ -z "$bucket_name" ]]; then
    errecho "ERROR: You must provide a bucket name with the -b parameter."
    usage
    return 1
  fi

  local bucket_config_arg
  # A location constraint for "us-east-1" returns an error.
  if [[ -n "$region_code" ]] && [[ "$region_code" != "us-east-1" ]]; then
    bucket_config_arg="--create-bucket-configuration LocationConstraint=
$region_code"
  fi
```

```
  iecho "Parameters:\n"
  iecho "    Bucket name:   $bucket_name"
  iecho "    Region code:   $region_code"
  iecho ""

  # If the bucket already exists, we don't want to try to create it.
  if (bucket_exists "$bucket_name"); then
    errecho "ERROR: A bucket with that name already exists. Try again."
    return 1
  fi

  # shellcheck disable=SC2086
  response=$(aws s3api create-bucket \
    --bucket "$bucket_name" \
    $bucket_config_arg)

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports create-bucket operation failed.\n$response"
    return 1
  fi
}

###############################################################################
# function copy_file_to_bucket
#
# This function creates a file in the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket to copy the file to.
#       $2 - The path and file name of the local file to copy to the bucket.
#       $3 - The key (name) to call the copy of the file in the bucket.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function copy_file_to_bucket() {
  local response bucket_name source_file destination_file_name
  bucket_name=$1
  source_file=$2
  destination_file_name=$3

  response=$(aws s3api put-object \
```

```
    --bucket "$bucket_name" \
    --body "$source_file" \
    --key "$destination_file_name")

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports put-object operation failed.\n$response"
    return 1
  fi
}

################################################################################
# function download_object_from_bucket
#
# This function downloads an object in a bucket to a file.
#
# Parameters:
#       $1 - The name of the bucket to download the object from.
#       $2 - The path and file name to store the downloaded bucket.
#       $3 - The key (name) of the object in the bucket.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function download_object_from_bucket() {
  local bucket_name=$1
  local destination_file_name=$2
  local object_name=$3
  local response

  response=$(aws s3api get-object \
    --bucket "$bucket_name" \
    --key "$object_name" \
    "$destination_file_name")

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    errecho "ERROR: AWS reports put-object operation failed.\n$response"
    return 1
  fi
}

################################################################################
```

```
# function copy_item_in_bucket
#
# This function creates a copy of the specified file in the same bucket.
#
# Parameters:
#       $1 - The name of the bucket to copy the file from and to.
#       $2 - The key of the source file to copy.
#       $3 - The key of the destination file.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function copy_item_in_bucket() {
  local bucket_name=$1
  local source_key=$2
  local destination_key=$3
  local response

  response=$(aws s3api copy-object \
    --bucket "$bucket_name" \
    --copy-source "$bucket_name/$source_key" \
    --key "$destination_key")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api copy-object operation failed.\n$response"
    return 1
  fi
}

###############################################################################
# function list_items_in_bucket
#
# This function displays a list of the files in the bucket with each file's
# size. The function uses the --query parameter to retrieve only the key and
# size fields from the Contents collection.
#
# Parameters:
#       $1 - The name of the bucket.
#
# Returns:
#       The list of files in text format.
#     And:
```

```
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function list_items_in_bucket() {
  local bucket_name=$1
  local response

  response=$(aws s3api list-objects \
    --bucket "$bucket_name" \
    --output text \
    --query 'Contents[].{Key: Key, Size: Size}')

  # shellcheck disable=SC2181
  if [[ ${?} -eq 0 ]]; then
    echo "$response"
  else
    errecho "ERROR: AWS reports s3api list-objects operation failed.\n$response"
    return 1
  fi
}

##############################################################################
# function delete_items_in_bucket
#
# This function deletes the specified list of keys from the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.
#       $2 - A list of keys in the bucket to delete.

# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function delete_items_in_bucket() {
  local bucket_name=$1
  local keys=$2
  local response

  # Create the JSON for the items to delete.
  local delete_items
  delete_items="{\"Objects\":["
  for key in $keys; do
    delete_items="$delete_items{\"Key\": \"$key\"},"
```

```
  done
  delete_items=${delete_items%?} # Remove the final comma.
  delete_items="$delete_items]}"

  response=$(aws s3api delete-objects \
    --bucket "$bucket_name" \
    --delete "$delete_items")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR:  AWS reports s3api delete-object operation failed.\n
$response"
    return 1
  fi
}

##############################################################################
# function delete_bucket
#
# This function deletes the specified bucket.
#
# Parameters:
#       $1 - The name of the bucket.

# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function delete_bucket() {
  local bucket_name=$1
  local response

  response=$(aws s3api delete-bucket \
    --bucket "$bucket_name")

  # shellcheck disable=SC2181
  if [[ $? -ne 0 ]]; then
    errecho "ERROR: AWS reports s3api delete-bucket failed.\n$response"
    return 1
  fi
}
```

- Untuk detail API, lihat topik berikut di Referensi Perintah AWS CLI .

- CopyObject
- CreateBucket
- DeleteBucket
- DeleteObjects
- GetObject
- ListObjectsV2
- PutObject

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
#include <iostream>
#include <aws/core/Aws.h>
#include <aws/s3/S3Client.h>
#include <aws/s3/model/CopyObjectRequest.h>
#include <aws/s3/model/CreateBucketRequest.h>
#include <aws/s3/model/DeleteBucketRequest.h>
#include <aws/s3/model/DeleteObjectRequest.h>
#include <aws/s3/model/GetObjectRequest.h>
#include <aws/s3/model/ListObjectsV2Request.h>
#include <aws/s3/model/PutObjectRequest.h>
#include <aws/s3/model/BucketLocationConstraint.h>
#include <aws/s3/model/CreateBucketConfiguration.h>
#include <aws/core/utils/UUID.h>
#include <aws/core/utils/StringUtils.h>
#include <aws/core/utils/memory/stl/AWSAllocator.h>
#include <fstream>
#include "s3_examples.h"

namespace AwsDoc {
    namespace S3 {
```

```
        //! Delete an S3 bucket.
        /*!
          \param bucketName: The S3 bucket's name.
          \param client: An S3 client.
          \return bool: Function succeeded.
        */
        static bool
        deleteBucket(const Aws::String &bucketName, Aws::S3::S3Client &client);

        //! Delete an object in an S3 bucket.
        /*!
          \param bucketName: The S3 bucket's name.
          \param key: The key for the object in the S3 bucket.
          \param client: An S3 client.
          \return bool: Function succeeded.
         */
        static bool
        deleteObjectFromBucket(const Aws::String &bucketName, const Aws::String
 &key,
                               Aws::S3::S3Client &client);
    }
}


//! Scenario to create, copy, and delete S3 buckets and objects.
/*!
  \param uploadFilePath: Path to file to upload to an Amazon S3 bucket.
  \param saveFilePath: Path for saving a downloaded S3 object.
  \param clientConfig: Aws client configuration.
  \return bool: Function succeeded.
 */
bool AwsDoc::S3::S3_GettingStartedScenario(const Aws::String &uploadFilePath,
                                           const Aws::String &saveFilePath,
                                           const Aws::Client::ClientConfiguration
 &clientConfig) {

    Aws::S3::S3Client client(clientConfig);

    // Create a unique bucket name which is only temporary and will be deleted.
    // Format: "doc-example-bucket-" + lowercase UUID.
    Aws::String uuid = Aws::Utils::UUID::RandomUUID();
    Aws::String bucketName = "doc-example-bucket-" +
                             Aws::Utils::StringUtils::ToLower(uuid.c_str());
```

```
   // 1. Create a bucket.
   {
       Aws::S3::Model::CreateBucketRequest request;
       request.SetBucket(bucketName);

       if (clientConfig.region != Aws::Region::US_EAST_1) {
           Aws::S3::Model::CreateBucketConfiguration createBucketConfiguration;
           createBucketConfiguration.WithLocationConstraint(

Aws::S3::Model::BucketLocationConstraintMapper::GetBucketLocationConstraintForName(
                           clientConfig.region));
           request.WithCreateBucketConfiguration(createBucketConfiguration);
       }

       Aws::S3::Model::CreateBucketOutcome outcome =
client.CreateBucket(request);

       if (!outcome.IsSuccess()) {
           const Aws::S3::S3Error &err = outcome.GetError();
           std::cerr << "Error: createBucket: " <<
                     err.GetExceptionName() << ": " << err.GetMessage() <<
std::endl;
           return false;
       } else {
           std::cout << "Created the bucket, '" << bucketName <<
                     "', in the region, '" << clientConfig.region << "'." <<
std::endl;
       }
   }

   // 2. Upload a local file to the bucket.
   Aws::String key = "key-for-test";
   {
       Aws::S3::Model::PutObjectRequest request;
       request.SetBucket(bucketName);
       request.SetKey(key);

       std::shared_ptr<Aws::FStream> input_data =
               Aws::MakeShared<Aws::FStream>("SampleAllocationTag",
                                             uploadFilePath,
                                             std::ios_base::in |
                                             std::ios_base::binary);

       if (!input_data->is_open()) {
```

```
            std::cerr << "Error: unable to open file, '" << uploadFilePath <<
"'."
                      << std::endl;
            AwsDoc::S3::deleteBucket(bucketName, client);
            return false;
        }

        request.SetBody(input_data);

        Aws::S3::Model::PutObjectOutcome outcome =
                client.PutObject(request);

        if (!outcome.IsSuccess()) {
            std::cerr << "Error: putObject: " <<
                      outcome.GetError().GetMessage() << std::endl;
            AwsDoc::S3::deleteObjectFromBucket(bucketName, key, client);
            AwsDoc::S3::deleteBucket(bucketName, client);
            return false;
        } else {
            std::cout << "Added the object with the key, '" << key
                      << "', to the bucket, '"
                      << bucketName << "'." << std::endl;
        }
    }

    // 3. Download the object to a local file.
    {
        Aws::S3::Model::GetObjectRequest request;
        request.SetBucket(bucketName);
        request.SetKey(key);

        Aws::S3::Model::GetObjectOutcome outcome =
                client.GetObject(request);

        if (!outcome.IsSuccess()) {
            const Aws::S3::S3Error &err = outcome.GetError();
            std::cerr << "Error: getObject: " <<
                      err.GetExceptionName() << ": " << err.GetMessage() <<
std::endl;
        } else {
            std::cout << "Downloaded the object with the key, '" << key
                      << "', in the bucket, '"
                      << bucketName << "'." << std::endl;
```

```
            Aws::IOStream &ioStream = outcome.GetResultWithOwnership().
                    GetBody();
            Aws::OFStream outStream(saveFilePath,
                                    std::ios_base::out | std::ios_base::binary);
            if (!outStream.is_open()) {
                std::cout << "Error: unable to open file, '" << saveFilePath <<
 "'."
                          << std::endl;
            } else {
                outStream << ioStream.rdbuf();
                std::cout << "Wrote the downloaded object to the file '"
                          << saveFilePath << "'." << std::endl;
            }
        }
    }

    // 4. Copy the object to a different "folder" in the bucket.
    Aws::String copiedToKey = "test-folder/" + key;
    {
        Aws::S3::Model::CopyObjectRequest request;
        request.WithBucket(bucketName)
                .WithKey(copiedToKey)
                .WithCopySource(bucketName + "/" + key);

        Aws::S3::Model::CopyObjectOutcome outcome =
                client.CopyObject(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error: copyObject: " <<
                      outcome.GetError().GetMessage() << std::endl;
        } else {
            std::cout << "Copied the object with the key, '" << key
                      << "', to the key, '" << copiedToKey
                      << ", in the bucket, '" << bucketName << "'." << std::endl;
        }
    }

    // 5. List objects in the bucket.
    {
        Aws::S3::Model::ListObjectsV2Request request;
        request.WithBucket(bucketName);

        Aws::String continuationToken;
        Aws::Vector<Aws::S3::Model::Object> allObjects;
```

```
        do {
            if (!continuationToken.empty()) {
                request.SetContinuationToken(continuationToken);
            }
            Aws::S3::Model::ListObjectsV2Outcome outcome = client.ListObjectsV2(
                    request);

            if (!outcome.IsSuccess()) {
                std::cerr << "Error: ListObjects: " <<
                          outcome.GetError().GetMessage() << std::endl;
                break;
            } else {
                Aws::Vector<Aws::S3::Model::Object> objects =
                        outcome.GetResult().GetContents();
                allObjects.insert(allObjects.end(), objects.begin(),
 objects.end());
                continuationToken = outcome.GetResult().GetContinuationToken();
            }
        } while (!continuationToken.empty());

        std::cout << allObjects.size() << " objects in the bucket, '" <<
 bucketName
                  << "':" << std::endl;

        for (Aws::S3::Model::Object &object: allObjects) {
            std::cout << "     '" << object.GetKey() << "'" << std::endl;
        }
    }

    // 6. Delete all objects in the bucket.
    // All objects in the bucket must be deleted before deleting the bucket.
    AwsDoc::S3::deleteObjectFromBucket(bucketName, copiedToKey, client);
    AwsDoc::S3::deleteObjectFromBucket(bucketName, key, client);

    // 7. Delete the bucket.
    return AwsDoc::S3::deleteBucket(bucketName, client);
}

bool AwsDoc::S3::deleteObjectFromBucket(const Aws::String &bucketName,
                                        const Aws::String &key,
                                        Aws::S3::S3Client &client) {
    Aws::S3::Model::DeleteObjectRequest request;
    request.SetBucket(bucketName);
    request.SetKey(key);
```

```
    Aws::S3::Model::DeleteObjectOutcome outcome =
            client.DeleteObject(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: deleteObject: " <<
                  outcome.GetError().GetMessage() << std::endl;
    } else {
        std::cout << "Deleted the object with the key, '" << key
                  << "', from the bucket, '"
                  << bucketName << "'." << std::endl;
    }

    return outcome.IsSuccess();
}

bool
AwsDoc::S3::deleteBucket(const Aws::String &bucketName, Aws::S3::S3Client
 &client) {
    Aws::S3::Model::DeleteBucketRequest request;
    request.SetBucket(bucketName);

    Aws::S3::Model::DeleteBucketOutcome outcome =
            client.DeleteBucket(request);

    if (!outcome.IsSuccess()) {
        const Aws::S3::S3Error &err = outcome.GetError();
        std::cerr << "Error: deleteBucket: " <<
                  err.GetExceptionName() << ": " << err.GetMessage() <<
 std::endl;
    } else {
        std::cout << "Deleted the bucket, '" << bucketName << "'." << std::endl;
    }
    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for C++ .
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects

- GetObject
- ListObjectsV2
- PutObject

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Tentukan struktur yang membungkus tindakan bucket dan objek yang digunakan oleh skenario.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// ListBuckets lists the buckets in the current account.
func (basics BucketBasics) ListBuckets() ([]types.Bucket, error) {
 result, err := basics.S3Client.ListBuckets(context.TODO(),
 &s3.ListBucketsInput{})
 var buckets []types.Bucket
 if err != nil {
  log.Printf("Couldn't list buckets for your account. Here's why: %v\n", err)
 } else {
  buckets = result.Buckets
 }
 return buckets, err
```

```
}



// BucketExists checks whether a bucket exists in the current account.
func (basics BucketBasics) BucketExists(bucketName string) (bool, error) {
 _, err := basics.S3Client.HeadBucket(context.TODO(), &s3.HeadBucketInput{
  Bucket: aws.String(bucketName),
 })
 exists := true
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NotFound:
    log.Printf("Bucket %v is available.\n", bucketName)
    exists = false
    err = nil
   default:
    log.Printf("Either you don't have access to bucket %v or another error
 occurred. "+
     "Here's what happened: %v\n", bucketName, err)
   }
  }
 } else {
  log.Printf("Bucket %v exists and you already own it.", bucketName)
 }

 return exists, err
}



// CreateBucket creates a bucket with the specified name in the specified Region.
func (basics BucketBasics) CreateBucket(name string, region string) error {
 _, err := basics.S3Client.CreateBucket(context.TODO(), &s3.CreateBucketInput{
  Bucket: aws.String(name),
  CreateBucketConfiguration: &types.CreateBucketConfiguration{
   LocationConstraint: types.BucketLocationConstraint(region),
  },
 })
 if err != nil {
  log.Printf("Couldn't create bucket %v in Region %v. Here's why: %v\n",
   name, region, err)
```

```
 }
 return err
}



// UploadFile reads from a file and puts the data into an object in a bucket.
func (basics BucketBasics) UploadFile(bucketName string, objectKey string,
 fileName string) error {
 file, err := os.Open(fileName)
 if err != nil {
  log.Printf("Couldn't open file %v to upload. Here's why: %v\n", fileName, err)
 } else {
  defer file.Close()
  _, err = basics.S3Client.PutObject(context.TODO(), &s3.PutObjectInput{
   Bucket: aws.String(bucketName),
   Key:    aws.String(objectKey),
   Body:   file,
  })
  if err != nil {
   log.Printf("Couldn't upload file %v to %v:%v. Here's why: %v\n",
    fileName, bucketName, objectKey, err)
  }
 }
 return err
}



// UploadLargeObject uses an upload manager to upload data to an object in a
 bucket.
// The upload manager breaks large data into parts and uploads the parts
 concurrently.
func (basics BucketBasics) UploadLargeObject(bucketName string, objectKey string,
 largeObject []byte) error {
 largeBuffer := bytes.NewReader(largeObject)
 var partMiBs int64 = 10
 uploader := manager.NewUploader(basics.S3Client, func(u *manager.Uploader) {
  u.PartSize = partMiBs * 1024 * 1024
 })
 _, err := uploader.Upload(context.TODO(), &s3.PutObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
  Body:   largeBuffer,
```

```
 })
 if err != nil {
  log.Printf("Couldn't upload large object to %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }

 return err
}



// DownloadFile gets an object from a bucket and stores it in a local file.
func (basics BucketBasics) DownloadFile(bucketName string, objectKey string,
 fileName string) error {
 result, err := basics.S3Client.GetObject(context.TODO(), &s3.GetObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't get object %v:%v. Here's why: %v\n", bucketName,
 objectKey, err)
  return err
 }
 defer result.Body.Close()
 file, err := os.Create(fileName)
 if err != nil {
  log.Printf("Couldn't create file %v. Here's why: %v\n", fileName, err)
  return err
 }
 defer file.Close()
 body, err := io.ReadAll(result.Body)
 if err != nil {
  log.Printf("Couldn't read object body from %v. Here's why: %v\n", objectKey,
 err)
 }
 _, err = file.Write(body)
 return err
}



// DownloadLargeObject uses a download manager to download an object from a
 bucket.
```

```
// The download manager gets the data in parts and writes them to a buffer until
 all of
// the data has been downloaded.
func (basics BucketBasics) DownloadLargeObject(bucketName string, objectKey
 string) ([]byte, error) {
 var partMiBs int64 = 10
 downloader := manager.NewDownloader(basics.S3Client, func(d *manager.Downloader)
 {
  d.PartSize = partMiBs * 1024 * 1024
 })
 buffer := manager.NewWriteAtBuffer([]byte{})
 _, err := downloader.Download(context.TODO(), buffer, &s3.GetObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't download large object from %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }
 return buffer.Bytes(), err
}



// CopyToFolder copies an object in a bucket to a subfolder in the same bucket.
func (basics BucketBasics) CopyToFolder(bucketName string, objectKey string,
 folderName string) error {
 _, err := basics.S3Client.CopyObject(context.TODO(), &s3.CopyObjectInput{
  Bucket:     aws.String(bucketName),
  CopySource: aws.String(fmt.Sprintf("%v/%v", bucketName, objectKey)),
  Key:        aws.String(fmt.Sprintf("%v/%v", folderName, objectKey)),
 })
 if err != nil {
  log.Printf("Couldn't copy object from %v:%v to %v:%v/%v. Here's why: %v\n",
   bucketName, objectKey, bucketName, folderName, objectKey, err)
 }
 return err
}



// CopyToBucket copies an object in a bucket to another bucket.
func (basics BucketBasics) CopyToBucket(sourceBucket string, destinationBucket
 string, objectKey string) error {
```

```
 _, err := basics.S3Client.CopyObject(context.TODO(), &s3.CopyObjectInput{
  Bucket:     aws.String(destinationBucket),
  CopySource: aws.String(fmt.Sprintf("%v/%v", sourceBucket, objectKey)),
  Key:        aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't copy object from %v:%v to %v:%v. Here's why: %v\n",
   sourceBucket, objectKey, destinationBucket, objectKey, err)
 }
 return err
}



// ListObjects lists the objects in a bucket.
func (basics BucketBasics) ListObjects(bucketName string) ([]types.Object, error)
 {
 result, err := basics.S3Client.ListObjectsV2(context.TODO(),
 &s3.ListObjectsV2Input{
  Bucket: aws.String(bucketName),
 })
 var contents []types.Object
 if err != nil {
  log.Printf("Couldn't list objects in bucket %v. Here's why: %v\n", bucketName,
 err)
 } else {
  contents = result.Contents
 }
 return contents, err
}



// DeleteObjects deletes a list of objects from a bucket.
func (basics BucketBasics) DeleteObjects(bucketName string, objectKeys []string)
 error {
 var objectIds []types.ObjectIdentifier
 for _, key := range objectKeys {
  objectIds = append(objectIds, types.ObjectIdentifier{Key: aws.String(key)})
 }
 output, err := basics.S3Client.DeleteObjects(context.TODO(),
 &s3.DeleteObjectsInput{
  Bucket: aws.String(bucketName),
  Delete: &types.Delete{Objects: objectIds},
```

```
 })
 if err != nil {
  log.Printf("Couldn't delete objects from bucket %v. Here's why: %v\n",
 bucketName, err)
 } else {
  log.Printf("Deleted %v objects.\n", len(output.Deleted))
 }
 return err
}



// DeleteBucket deletes a bucket. The bucket must be empty or an error is
 returned.
func (basics BucketBasics) DeleteBucket(bucketName string) error {
 _, err := basics.S3Client.DeleteBucket(context.TODO(), &s3.DeleteBucketInput{
  Bucket: aws.String(bucketName)})
 if err != nil {
  log.Printf("Couldn't delete bucket %v. Here's why: %v\n", bucketName, err)
 }
 return err
}
```

Jalankan skenario interaktif yang menunjukkan cara kerja dengan bucket dan objek S3.

```
// RunGetStartedScenario is an interactive example that shows you how to use
 Amazon
// Simple Storage Service (Amazon S3) to create an S3 bucket and use it to store
 objects.
//
// 1. Create a bucket.
// 2. Upload a local file to the bucket.
// 3. Upload a large object to the bucket by using an upload manager.
// 4. Download an object to a local file.
// 5. Download a large object by using a download manager.
// 6. Copy an object to a different folder in the bucket.
// 7. List objects in the bucket.
// 8. Delete all objects in the bucket.
// 9. Delete the bucket.
//
```

```
// This example creates an Amazon S3 service client from the specified sdkConfig
 so that
// you can replace it with a mocked or stubbed config for unit testing.
//
// It uses a questioner from the `demotools` package to get input during the
 example.
// This package can be found in the ..\..\demotools folder of this repo.
func RunGetStartedScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner) {
 defer func() {
  if r := recover(); r != nil {
   fmt.Println("Something went wrong with the demo.\n", r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the Amazon S3 getting started demo.")
 log.Println(strings.Repeat("-", 88))

 s3Client := s3.NewFromConfig(sdkConfig)
 bucketBasics := actions.BucketBasics{S3Client: s3Client}

 count := 10
 log.Printf("Let's list up to %v buckets for your account:", count)
 buckets, err := bucketBasics.ListBuckets()
 if err != nil {
  panic(err)
 }
 if len(buckets) == 0 {
  log.Println("You don't have any buckets!")
 } else {
  if count > len(buckets) {
   count = len(buckets)
  }
  for _, bucket := range buckets[:count] {
   log.Printf("\t%v\n", *bucket.Name)
  }
 }

 bucketName := questioner.Ask("Let's create a bucket. Enter a name for your
 bucket:",
  demotools.NotEmpty{})
 bucketExists, err := bucketBasics.BucketExists(bucketName)
 if err != nil {
```

```
 panic(err)
}
if !bucketExists {
 err = bucketBasics.CreateBucket(bucketName, sdkConfig.Region)
 if err != nil {
  panic(err)
 } else {
  log.Println("Bucket created.")
 }
}
log.Println(strings.Repeat("-", 88))

fmt.Println("Let's upload a file to your bucket.")
smallFile := questioner.Ask("Enter the path to a file you want to upload:",
 demotools.NotEmpty{})
const smallKey = "doc-example-key"
err = bucketBasics.UploadFile(bucketName, smallKey, smallFile)
if err != nil {
 panic(err)
}
log.Printf("Uploaded %v as %v.\n", smallFile, smallKey)
log.Println(strings.Repeat("-", 88))

mibs := 30
log.Printf("Let's create a slice of %v MiB of random bytes and upload it to your
bucket. ", mibs)
questioner.Ask("Press Enter when you're ready.")
largeBytes := make([]byte, 1024*1024*mibs)
rand.Seed(time.Now().Unix())
rand.Read(largeBytes)
largeKey := "doc-example-large"
log.Println("Uploading...")
err = bucketBasics.UploadLargeObject(bucketName, largeKey, largeBytes)
if err != nil {
 panic(err)
}
log.Printf("Uploaded %v MiB object as %v", mibs, largeKey)
log.Println(strings.Repeat("-", 88))

log.Printf("Let's download %v to a file.", smallKey)
downloadFileName := questioner.Ask("Enter a name for the downloaded file:",
demotools.NotEmpty{})
err = bucketBasics.DownloadFile(bucketName, smallKey, downloadFileName)
if err != nil {
```

```
 panic(err)
}
log.Printf("File %v downloaded.", downloadFileName)
log.Println(strings.Repeat("-", 88))

log.Printf("Let's download the %v MiB object.", mibs)
questioner.Ask("Press Enter when you're ready.")
log.Println("Downloading...")
largeDownload, err := bucketBasics.DownloadLargeObject(bucketName, largeKey)
if err != nil {
 panic(err)
}
log.Printf("Downloaded %v bytes.", len(largeDownload))
log.Println(strings.Repeat("-", 88))

log.Printf("Let's copy %v to a folder in the same bucket.", smallKey)
folderName := questioner.Ask("Enter a folder name: ", demotools.NotEmpty{})
err = bucketBasics.CopyToFolder(bucketName, smallKey, folderName)
if err != nil {
 panic(err)
}
log.Printf("Copied %v to %v/%v.\n", smallKey, folderName, smallKey)
log.Println(strings.Repeat("-", 88))

log.Println("Let's list the objects in your bucket.")
questioner.Ask("Press Enter when you're ready.")
objects, err := bucketBasics.ListObjects(bucketName)
if err != nil {
 panic(err)
}
log.Printf("Found %v objects.\n", len(objects))
var objKeys []string
for _, object := range objects {
 objKeys = append(objKeys, *object.Key)
 log.Printf("\t%v\n", *object.Key)
}
log.Println(strings.Repeat("-", 88))

if questioner.AskBool("Do you want to delete your bucket and all of its "+
 "contents? (y/n)", "y") {
 log.Println("Deleting objects.")
 err = bucketBasics.DeleteObjects(bucketName, objKeys)
 if err != nil {
  panic(err)
```

```
  }
  log.Println("Deleting bucket.")
  err = bucketBasics.DeleteBucket(bucketName)
  if err != nil {
   panic(err)
  }
  log.Printf("Deleting downloaded file %v.\n", downloadFileName)
  err = os.Remove(downloadFileName)
  if err != nil {
   panic(err)
  }
 } else {
  log.Println("Okay. Don't forget to delete objects from your bucket to avoid
 charges.")
 }
 log.Println(strings.Repeat("-", 88))

 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Go .
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * This Java code example performs the following tasks:
 *
 * 1. Creates an Amazon S3 bucket.
 * 2. Uploads an object to the bucket.
 * 3. Downloads the object to another local file.
 * 4. Uploads an object using multipart upload.
 * 5. List all objects located in the Amazon S3 bucket.
 * 6. Copies the object to another Amazon S3 bucket.
 * 7. Deletes the object from the Amazon S3 bucket.
 * 8. Deletes the Amazon S3 bucket.
 */

public class S3Scenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws IOException {
        final String usage = """

                Usage:
                    <bucketName> <key> <objectPath> <savePath> <toBucket>

                Where:
```

```
                    bucketName - The Amazon S3 bucket to create.
                    key - The key to use.
                    objectPath - The path where the file is located (for example,
C:/AWS/book2.pdf).
                    savePath - The path where the file is saved after it's
downloaded (for example, C:/AWS/book2.pdf).
                    toBucket - An Amazon S3 bucket to where an object is copied
to (for example, C:/AWS/book2.pdf).\s
                    """;

        if (args.length != 5) {
            System.out.println(usage);
            System.exit(1);
        }

        String bucketName = args[0];
        String key = args[1];
        String objectPath = args[2];
        String savePath = args[3];
        String toBucket = args[4];
        Region region = Region.US_EAST_1;
        S3Client s3 = S3Client.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the Amazon S3 example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("1. Create an Amazon S3 bucket.");
        createBucket(s3, bucketName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("2. Update a local file to the Amazon S3 bucket.");
        uploadLocalFile(s3, bucketName, key, objectPath);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Download the object to another local file.");
        getObjectBytes(s3, bucketName, key, savePath);
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("4. Perform a multipart upload.");
        String multipartKey = "multiPartKey";
        multipartUpload(s3, toBucket, multipartKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("5. List all objects located in the Amazon S3
bucket.");
        listAllObjects(s3, bucketName);
        anotherListExample(s3, bucketName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6. Copy the object to another Amazon S3 bucket.");
        copyBucketObject(s3, bucketName, key, toBucket);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Delete the object from the Amazon S3 bucket.");
        deleteObjectFromBucket(s3, bucketName, key);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Delete the Amazon S3 bucket.");
        deleteBucket(s3, bucketName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("All Amazon S3 operations were successfully
performed");
        System.out.println(DASHES);
        s3.close();
    }

    // Create a bucket by using a S3Waiter object.
    public static void createBucket(S3Client s3Client, String bucketName) {
        try {
            S3Waiter s3Waiter = s3Client.waiter();
            CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
                    .bucket(bucketName)
                    .build();

            s3Client.createBucket(bucketRequest);
```

```
            HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
                    .bucket(bucketName)
                    .build();

            // Wait until the bucket is created and print out the response.
            WaiterResponse<HeadBucketResponse> waiterResponse =
s3Waiter.waitUntilBucketExists(bucketRequestWait);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println(bucketName + " is ready");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteBucket(S3Client client, String bucket) {
        DeleteBucketRequest deleteBucketRequest = DeleteBucketRequest.builder()
                .bucket(bucket)
                .build();

        client.deleteBucket(deleteBucketRequest);
        System.out.println(bucket + " was deleted.");
    }

    /**
     * Upload an object in parts.
     */
    public static void multipartUpload(S3Client s3, String bucketName, String
key) {
        int mB = 1024 * 1024;
        // First create a multipart upload and get the upload id.
        CreateMultipartUploadRequest createMultipartUploadRequest =
CreateMultipartUploadRequest.builder()
                .bucket(bucketName)
                .key(key)
                .build();

        CreateMultipartUploadResponse response =
s3.createMultipartUpload(createMultipartUploadRequest);
        String uploadId = response.uploadId();
        System.out.println(uploadId);

        // Upload all the different parts of the object.
```

```
        UploadPartRequest uploadPartRequest1 = UploadPartRequest.builder()
                .bucket(bucketName)
                .key(key)
                .uploadId(uploadId)
                .partNumber(1).build();

        String etag1 = s3.uploadPart(uploadPartRequest1,
RequestBody.fromByteBuffer(getRandomByteBuffer(5 * mB)))
                .eTag();
        CompletedPart part1 =
CompletedPart.builder().partNumber(1).eTag(etag1).build();

        UploadPartRequest uploadPartRequest2 =
UploadPartRequest.builder().bucket(bucketName).key(key)
                .uploadId(uploadId)
                .partNumber(2).build();
        String etag2 = s3.uploadPart(uploadPartRequest2,
RequestBody.fromByteBuffer(getRandomByteBuffer(3 * mB)))
                .eTag();
        CompletedPart part2 =
CompletedPart.builder().partNumber(2).eTag(etag2).build();

        // Call completeMultipartUpload operation to tell S3 to merge all
uploaded
        // parts and finish the multipart operation.
        CompletedMultipartUpload completedMultipartUpload =
CompletedMultipartUpload.builder()
                .parts(part1, part2)
                .build();

        CompleteMultipartUploadRequest completeMultipartUploadRequest =
CompleteMultipartUploadRequest.builder()
                .bucket(bucketName)
                .key(key)
                .uploadId(uploadId)
                .multipartUpload(completedMultipartUpload)
                .build();

        s3.completeMultipartUpload(completeMultipartUploadRequest);
    }

    private static ByteBuffer getRandomByteBuffer(int size) {
        byte[] b = new byte[size];
        new Random().nextBytes(b);
```

```
        return ByteBuffer.wrap(b);
    }

    public static void getObjectBytes(S3Client s3, String bucketName, String
keyName, String path) {
        try {
            GetObjectRequest objectRequest = GetObjectRequest
                    .builder()
                    .key(keyName)
                    .bucket(bucketName)
                    .build();

            ResponseBytes<GetObjectResponse> objectBytes =
s3.getObjectAsBytes(objectRequest);
            byte[] data = objectBytes.asByteArray();

            // Write the data to a local file.
            File myFile = new File(path);
            OutputStream os = new FileOutputStream(myFile);
            os.write(data);
            System.out.println("Successfully obtained bytes from an S3 object");
            os.close();

        } catch (IOException ex) {
            ex.printStackTrace();
        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void uploadLocalFile(S3Client s3, String bucketName, String
key, String objectPath) {
        PutObjectRequest objectRequest = PutObjectRequest.builder()
                .bucket(bucketName)
                .key(key)
                .build();

        s3.putObject(objectRequest, RequestBody.fromFile(new File(objectPath)));
    }

    public static void listAllObjects(S3Client s3, String bucketName) {
        ListObjectsV2Request listObjectsReqManual =
ListObjectsV2Request.builder()
```

```
                .bucket(bucketName)
                .maxKeys(1)
                .build();

        boolean done = false;
        while (!done) {
            ListObjectsV2Response listObjResponse =
s3.listObjectsV2(listObjectsReqManual);
            for (S3Object content : listObjResponse.contents()) {
                System.out.println(content.key());
            }

            if (listObjResponse.nextContinuationToken() == null) {
                done = true;
            }

            listObjectsReqManual = listObjectsReqManual.toBuilder()
                    .continuationToken(listObjResponse.nextContinuationToken())
                    .build();
        }
    }

    public static void anotherListExample(S3Client s3, String bucketName) {
        ListObjectsV2Request listReq = ListObjectsV2Request.builder()
                .bucket(bucketName)
                .maxKeys(1)
                .build();

        ListObjectsV2Iterable listRes = s3.listObjectsV2Paginator(listReq);

        // Process response pages.
        listRes.stream()
                .flatMap(r -> r.contents().stream())
                .forEach(content -> System.out.println(" Key: " + content.key() +
" size = " + content.size()));

        // Helper method to work with paginated collection of items directly.
        listRes.contents().stream()
                .forEach(content -> System.out.println(" Key: " + content.key() +
" size = " + content.size()));

        for (S3Object content : listRes.contents()) {
            System.out.println(" Key: " + content.key() + " size = " +
content.size());
```

```
        }
    }

    public static void deleteObjectFromBucket(S3Client s3, String bucketName,
 String key) {
        DeleteObjectRequest deleteObjectRequest = DeleteObjectRequest.builder()
                .bucket(bucketName)
                .key(key)
                .build();

        s3.deleteObject(deleteObjectRequest);
        System.out.println(key + " was deleted");
    }

    public static String copyBucketObject(S3Client s3, String fromBucket, String
 objectKey, String toBucket) {
        String encodedUrl = null;
        try {
            encodedUrl = URLEncoder.encode(fromBucket + "/" + objectKey,
 StandardCharsets.UTF_8.toString());
        } catch (UnsupportedEncodingException e) {
            System.out.println("URL could not be encoded: " + e.getMessage());
        }
        CopyObjectRequest copyReq = CopyObjectRequest.builder()
                .copySource(encodedUrl)
                .destinationBucket(toBucket)
                .destinationKey(objectKey)
                .build();

        try {
            CopyObjectResponse copyRes = s3.copyObject(copyReq);
            System.out.println("The " + objectKey + " was copied to " +
 toBucket);
            return copyRes.copyObjectResult().toString();

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Java 2.x .
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Pertama, impor semua modul yang diperlukan.

```
// Used to check if currently running file is this file.
import { fileURLToPath } from "url";
import { readdirSync, readFileSync, writeFileSync } from "fs";

// Local helper utils.
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";
import { Prompter } from "@aws-doc-sdk-examples/lib/prompter.js";
import { wrapText } from "@aws-doc-sdk-examples/lib/utils/util-string.js";

import {
  S3Client,
  CreateBucketCommand,
  PutObjectCommand,
  ListObjectsCommand,
  CopyObjectCommand,
  GetObjectCommand,
  DeleteObjectsCommand,
  DeleteBucketCommand,
```

```
} from "@aws-sdk/client-s3";
```

Impor sebelumnya merujuk beberapa utilitas pembantu. Utilitas ini bersifat lokal ke GitHub repositori yang ditautkan di awal bagian ini. Untuk referensi, lihat implementasi utilitas tersebut berikut ini.

```
export const dirnameFromMetaUrl = (metaUrl) =>
  fileURLToPath(new URL(".", metaUrl));

import { select, input, confirm, checkbox } from "@inquirer/prompts";

export class Prompter {
  /**
   * @param {{ message: string, choices: { name: string, value: string }[]}}
 options
   */
  select(options) {
    return select(options);
  }

  /**
   * @param {{ message: string }} options
   */
  input(options) {
    return input(options);
  }

  /**
   * @param {string} prompt
   */
  checkContinue = async (prompt = "") => {
    const prefix = prompt && prompt + " ";
    let ok = await this.confirm({
      message: `${prefix}Continue?`,
    });
    if (!ok) throw new Error("Exiting...");
  };

  /**
   * @param {{ message: string }} options
   */
  confirm(options) {
```

```
    return confirm(options);
  }

  /**
   * @param {{ message: string, choices: { name: string, value: string }[]}}
 options
   */
  checkbox(options) {
    return checkbox(options);
  }
}

export const wrapText = (text, char = "=") => {
  const rule = char.repeat(80);
  return `${rule}\n    ${text}\n${rule}\n`;
};
```

Objek di S3 disimpan dalam 'bucket'. Mari kita tentukan fungsi untuk membuat bucket baru.

```
export const createBucket = async () => {
  const bucketName = await prompter.input({
    message: "Enter a bucket name. Bucket names must be globally unique:",
  });
  const command = new CreateBucketCommand({ Bucket: bucketName });
  await s3Client.send(command);
  console.log("Bucket created successfully.\n");
  return bucketName;
};
```

Bucket berisi 'objek'. Fungsi ini mengunggah isi direktori ke bucket Anda sebagai objek.

```
export const uploadFilesToBucket = async ({ bucketName, folderPath }) => {
  console.log(`Uploading files from ${folderPath}\n`);
  const keys = readdirSync(folderPath);
  const files = keys.map((key) => {
    const filePath = `${folderPath}/${key}`;
    const fileContent = readFileSync(filePath);
    return {
      Key: key,
      Body: fileContent,
    };
```

```
  });

  for (let file of files) {
    await s3Client.send(
      new PutObjectCommand({
        Bucket: bucketName,
        Body: file.Body,
        Key: file.Key,
      }),
    );
    console.log(`${file.Key} uploaded successfully.`);
  }
};
```

Setelah mengunggah objek, pastikan bahwa objek tersebut diunggah dengan benar. Anda dapat menggunakannya ListObjects untuk itu. Anda akan menggunakan properti 'Kunci', tetapi ada juga properti lain yang berguna dalam respons.

```
export const listFilesInBucket = async ({ bucketName }) => {
  const command = new ListObjectsCommand({ Bucket: bucketName });
  const { Contents } = await s3Client.send(command);
  const contentsList = Contents.map((c) => ` • ${c.Key}`).join("\n");
  console.log("\nHere's a list of files in the bucket:");
  console.log(contentsList + "\n");
};
```

Terkadang Anda perlu menyalin objek dari satu bucket ke bucket lainnya. Gunakan CopyObject perintah untuk itu.

```
export const copyFileFromBucket = async ({ destinationBucket }) => {
  const proceed = await prompter.confirm({
    message: "Would you like to copy an object from another bucket?",
  });

  if (!proceed) {
    return;
  } else {
    const copy = async () => {
      try {
        const sourceBucket = await prompter.input({
```

```
          message: "Enter source bucket name:",
        });
        const sourceKey = await prompter.input({
          message: "Enter source key:",
        });
        const destinationKey = await prompter.input({
          message: "Enter destination key:",
        });

        const command = new CopyObjectCommand({
          Bucket: destinationBucket,
          CopySource: `${sourceBucket}/${sourceKey}`,
          Key: destinationKey,
        });
        await s3Client.send(command);
        await copyFileFromBucket({ destinationBucket });
      } catch (err) {
        console.error(`Copy error.`);
        console.error(err);
        const retryAnswer = await prompter.confirm({ message: "Try again?" });
        if (retryAnswer) {
          await copy();
        }
      }
    };
    await copy();
  }
};
```

Tidak ada metode SDK yang bisa digunakan untuk mendapatkan beberapa objek dari sebuah bucket. Tetapi, Anda akan membuat daftar objek yang akan diunduh dan mengulanginya.

```
export const downloadFilesFromBucket = async ({ bucketName }) => {
  const { Contents } = await s3Client.send(
    new ListObjectsCommand({ Bucket: bucketName }),
  );
  const path = await prompter.input({
    message: "Enter destination path for files:",
  });

  for (let content of Contents) {
    const obj = await s3Client.send(
```

```
      new GetObjectCommand({ Bucket: bucketName, Key: content.Key }),
    );
    writeFileSync(
      `${path}/${content.Key}`,
      await obj.Body.transformToByteArray(),
    );
  }
  console.log("Files downloaded successfully.\n");
};
```

Saatnya membersihkan sumber daya Anda. Bucket harus kosong sebelum dapat dihapus. Kedua fungsi ini mengosongkan dan menghapus bucket.

```
export const emptyBucket = async ({ bucketName }) => {
  const listObjectsCommand = new ListObjectsCommand({ Bucket: bucketName });
  const { Contents } = await s3Client.send(listObjectsCommand);
  const keys = Contents.map((c) => c.Key);

  const deleteObjectsCommand = new DeleteObjectsCommand({
    Bucket: bucketName,
    Delete: { Objects: keys.map((key) => ({ Key: key })) },
  });
  await s3Client.send(deleteObjectsCommand);
  console.log(`${bucketName} emptied successfully.\n`);
};

export const deleteBucket = async ({ bucketName }) => {
  const command = new DeleteBucketCommand({ Bucket: bucketName });
  await s3Client.send(command);
  console.log(`${bucketName} deleted successfully.\n`);
};
```

Fungsi 'utama' menyatukan semuanya. Jika Anda menjalankan file ini secara langsung, fungsi utama akan dipanggil.

```
const main = async () => {
  const OBJECT_DIRECTORY = `${dirnameFromMetaUrl(
    import.meta.url,
  )}../../../../resources/sample_files/.sample_media`;

  try {
```

```
    console.log(wrapText("Welcome to the Amazon S3 getting started example."));
    console.log("Let's create a bucket.");
    const bucketName = await createBucket();
    await prompter.confirm({ message: continueMessage });

    console.log(wrapText("File upload."));
    console.log(
      "I have some default files ready to go. You can edit the source code to
 provide your own.",
    );
    await uploadFilesToBucket({
      bucketName,
      folderPath: OBJECT_DIRECTORY,
    });

    await listFilesInBucket({ bucketName });
    await prompter.confirm({ message: continueMessage });

    console.log(wrapText("Copy files."));
    await copyFileFromBucket({ destinationBucket: bucketName });
    await listFilesInBucket({ bucketName });
    await prompter.confirm({ message: continueMessage });

    console.log(wrapText("Download files."));
    await downloadFilesFromBucket({ bucketName });

    console.log(wrapText("Clean up."));
    await emptyBucket({ bucketName });
    await deleteBucket({ bucketName });
  } catch (err) {
    console.error(err);
  }
};
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for JavaScript .
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject

- ListObjectsV2
- PutObject

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <bucketName> <key> <objectPath> <savePath> <toBucket>

    Where:
        bucketName - The Amazon S3 bucket to create.
        key - The key to use.
        objectPath - The path where the file is located (for example, C:/AWS/
book2.pdf).
        savePath - The path where the file is saved after it's downloaded (for
 example, C:/AWS/book2.pdf).
        toBucket - An Amazon S3 bucket to where an object is copied to (for
 example, C:/AWS/book2.pdf).
        """

    if (args.size != 4) {
        println(usage)
        exitProcess(1)
    }

    val bucketName = args[0]
    val key = args[1]
    val objectPath = args[2]
    val savePath = args[3]
    val toBucket = args[4]

    // Create an Amazon S3 bucket.
```

```
    createBucket(bucketName)

    // Update a local file to the Amazon S3 bucket.
    putObject(bucketName, key, objectPath)

    // Download the object to another local file.
    getObjectFromMrap(bucketName, key, savePath)

    // List all objects located in the Amazon S3 bucket.
    listBucketObs(bucketName)

    // Copy the object to another Amazon S3 bucket
    copyBucketOb(bucketName, key, toBucket)

    // Delete the object from the Amazon S3 bucket.
    deleteBucketObs(bucketName, key)

    // Delete the Amazon S3 bucket.
    deleteBucket(bucketName)
    println("All Amazon S3 operations were successfully performed")
}

suspend fun createBucket(bucketName: String) {
    val request =
        CreateBucketRequest {
            bucket = bucketName
        }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.createBucket(request)
        println("$bucketName is ready")
    }
}

suspend fun putObject(
    bucketName: String,
    objectKey: String,
    objectPath: String,
) {
    val metadataVal = mutableMapOf<String, String>()
    metadataVal["myVal"] = "test"

    val request =
        PutObjectRequest {
```

```
            bucket = bucketName
            key = objectKey
            metadata = metadataVal
            this.body = Paths.get(objectPath).asByteStream()
        }

    S3Client { region = "us-east-1" }.use { s3 ->
        val response = s3.putObject(request)
        println("Tag information is ${response.eTag}")
    }
}

suspend fun getObjectFromMrap(
    bucketName: String,
    keyName: String,
    path: String,
) {
    val request =
        GetObjectRequest {
            key = keyName
            bucket = bucketName
        }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.getObject(request) { resp ->
            val myFile = File(path)
            resp.body?.writeToFile(myFile)
            println("Successfully read $keyName from $bucketName")
        }
    }
}

suspend fun listBucketObs(bucketName: String) {
    val request =
        ListObjectsRequest {
            bucket = bucketName
        }

    S3Client { region = "us-east-1" }.use { s3 ->

        val response = s3.listObjects(request)
        response.contents?.forEach { myObject ->
            println("The name of the key is ${myObject.key}")
            println("The owner is ${myObject.owner}")
```

```
        }
    }
}

suspend fun copyBucketOb(
    fromBucket: String,
    objectKey: String,
    toBucket: String,
) {
    var encodedUrl = ""
    try {
        encodedUrl = URLEncoder.encode("$fromBucket/$objectKey",
 StandardCharsets.UTF_8.toString())
    } catch (e: UnsupportedEncodingException) {
        println("URL could not be encoded: " + e.message)
    }

    val request =
        CopyObjectRequest {
            copySource = encodedUrl
            bucket = toBucket
            key = objectKey
        }
    S3Client { region = "us-east-1" }.use { s3 ->
        s3.copyObject(request)
    }
}

suspend fun deleteBucketObs(
    bucketName: String,
    objectName: String,
) {
    val objectId =
        ObjectIdentifier {
            key = objectName
        }

    val delOb =
        Delete {
            objects = listOf(objectId)
        }

    val request =
        DeleteObjectsRequest {
```

```
            bucket = bucketName
            delete = delOb
        }

    S3Client { region = "us-east-1" }.use { s3 ->
        s3.deleteObjects(request)
        println("$objectName was deleted from $bucketName")
    }
}

suspend fun deleteBucket(bucketName: String?) {
    val request =
        DeleteBucketRequest {
            bucket = bucketName
        }
    S3Client { region = "us-east-1" }.use { s3 ->
        s3.deleteBucket(request)
        println("The $bucketName was successfully deleted!")
    }
}
```

- Untuk detail API, lihat topik berikut di referensi API SDK untuk Kotlin AWS .
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        echo("\n");
        echo("--------------------------------------\n");
        print("Welcome to the Amazon S3 getting started demo using PHP!\n");
        echo("--------------------------------------\n");

        $region = 'us-west-2';

        $this->s3client = new S3Client([
                'region' => $region,
        ]);
        /* Inline declaration example
        $s3client = new Aws\S3\S3Client(['region' => 'us-west-2']);
        */

        $this->bucketName = "doc-example-bucket-" . uniqid();

        try {
            $this->s3client->createBucket([
                'Bucket' => $this->bucketName,
                'CreateBucketConfiguration' => ['LocationConstraint' => $region],
            ]);
            echo "Created bucket named: $this->bucketName \n";
        } catch (Exception $exception) {
            echo "Failed to create bucket $this->bucketName with error: " .
$exception->getMessage();
            exit("Please fix error with bucket creation before continuing.");
        }

        $fileName = __DIR__ . "/local-file-" . uniqid();
        try {
            $this->s3client->putObject([
                'Bucket' => $this->bucketName,
```

```
                'Key' => $fileName,
                'SourceFile' => __DIR__ . '/testfile.txt'
            ]);
            echo "Uploaded $fileName to $this->bucketName.\n";
        } catch (Exception $exception) {
            echo "Failed to upload $fileName with error: " . $exception-
>getMessage();
            exit("Please fix error with file upload before continuing.");
        }

        try {
            $file = $this->s3client->getObject([
                'Bucket' => $this->bucketName,
                'Key' => $fileName,
            ]);
            $body = $file->get('Body');
            $body->rewind();
            echo "Downloaded the file and it begins with: {$body->read(26)}.\n";
        } catch (Exception $exception) {
            echo "Failed to download $fileName from $this->bucketName with error:
 " . $exception->getMessage();
            exit("Please fix error with file downloading before continuing.");
        }

        try {
            $folder = "copied-folder";
            $this->s3client->copyObject([
                'Bucket' => $this->bucketName,
                'CopySource' => "$this->bucketName/$fileName",
                'Key' => "$folder/$fileName-copy",
            ]);
            echo "Copied $fileName to $folder/$fileName-copy.\n";
        } catch (Exception $exception) {
            echo "Failed to copy $fileName with error: " . $exception-
>getMessage();
            exit("Please fix error with object copying before continuing.");
        }

        try {
            $contents = $this->s3client->listObjectsV2([
                'Bucket' => $this->bucketName,
            ]);
            echo "The contents of your bucket are: \n";
            foreach ($contents['Contents'] as $content) {
```

```
                echo $content['Key'] . "\n";
            }
        } catch (Exception $exception) {
            echo "Failed to list objects in $this->bucketName with error: " .
 $exception->getMessage();
            exit("Please fix error with listing objects before continuing.");
        }

        try {
            $objects = [];
            foreach ($contents['Contents'] as $content) {
                $objects[] = [
                    'Key' => $content['Key'],
                ];
            }
            $this->s3client->deleteObjects([
                'Bucket' => $this->bucketName,
                'Delete' => [
                    'Objects' => $objects,
                ],
            ]);
            $check = $this->s3client->listObjectsV2([
                'Bucket' => $this->bucketName,
            ]);
            if (count($check) <= 0) {
                throw new Exception("Bucket wasn't empty.");
            }
            echo "Deleted all objects and folders from $this->bucketName.\n";
        } catch (Exception $exception) {
            echo "Failed to delete $fileName from $this->bucketName with error:
 " . $exception->getMessage();
            exit("Please fix error with object deletion before continuing.");
        }

        try {
            $this->s3client->deleteBucket([
                'Bucket' => $this->bucketName,
            ]);
            echo "Deleted bucket $this->bucketName.\n";
        } catch (Exception $exception) {
            echo "Failed to delete $this->bucketName with error: " . $exception-
>getMessage();
            exit("Please fix error with bucket deletion before continuing.");
        }
```

```
        echo "Successfully ran the Amazon S3 with PHP demo.\n";
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for PHP .
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import io
import os
import uuid

import boto3
from boto3.s3.transfer import S3UploadFailedError
from botocore.exceptions import ClientError


def do_scenario(s3_resource):
    print("-" * 88)
    print("Welcome to the Amazon S3 getting started demo!")
    print("-" * 88)
```

```
    bucket_name = f"doc-example-bucket-{uuid.uuid4()}"
    bucket = s3_resource.Bucket(bucket_name)
    try:
        bucket.create(
            CreateBucketConfiguration={
                "LocationConstraint": s3_resource.meta.client.meta.region_name
            }
        )
        print(f"Created demo bucket named {bucket.name}.")
    except ClientError as err:
        print(f"Tried and failed to create demo bucket {bucket_name}.")
        print(f"\t{err.response['Error']['Code']}:{err.response['Error']
['Message']}")
        print(f"\nCan't continue the demo without a bucket!")
        return

    file_name = None
    while file_name is None:
        file_name = input("\nEnter a file you want to upload to your bucket: ")
        if not os.path.exists(file_name):
            print(f"Couldn't find file {file_name}. Are you sure it exists?")
            file_name = None

    obj = bucket.Object(os.path.basename(file_name))
    try:
        obj.upload_file(file_name)
        print(
            f"Uploaded file {file_name} into bucket {bucket.name} with key
 {obj.key}."
        )
    except S3UploadFailedError as err:
        print(f"Couldn't upload file {file_name} to {bucket.name}.")
        print(f"\t{err}")

    answer = input(f"\nDo you want to download {obj.key} into memory (y/n)? ")
    if answer.lower() == "y":
        data = io.BytesIO()
        try:
            obj.download_fileobj(data)
            data.seek(0)
            print(f"Got your object. Here are the first 20 bytes:\n")
            print(f"\t{data.read(20)}")
        except ClientError as err:
            print(f"Couldn't download {obj.key}.")
```

```
            print(
                f"\t{err.response['Error']['Code']}:{err.response['Error']
['Message']}"
            )

    answer = input(
        f"\nDo you want to copy {obj.key} to a subfolder in your bucket (y/n)? "
    )
    if answer.lower() == "y":
        dest_obj = bucket.Object(f"demo-folder/{obj.key}")
        try:
            dest_obj.copy({"Bucket": bucket.name, "Key": obj.key})
            print(f"Copied {obj.key} to {dest_obj.key}.")
        except ClientError as err:
            print(f"Couldn't copy {obj.key} to {dest_obj.key}.")
            print(
                f"\t{err.response['Error']['Code']}:{err.response['Error']
['Message']}"
            )

    print("\nYour bucket contains the following objects:")
    try:
        for o in bucket.objects.all():
            print(f"\t{o.key}")
    except ClientError as err:
        print(f"Couldn't list the objects in bucket {bucket.name}.")
        print(f"\t{err.response['Error']['Code']}:{err.response['Error']
['Message']}")

    answer = input(
        "\nDo you want to delete all of the objects as well as the bucket (y/n)?
 "
    )
    if answer.lower() == "y":
        try:
            bucket.objects.delete()
            bucket.delete()
            print(f"Emptied and deleted bucket {bucket.name}.\n")
        except ClientError as err:
            print(f"Couldn't empty and delete bucket {bucket.name}.")
            print(
                f"\t{err.response['Error']['Code']}:{err.response['Error']
['Message']}"
            )
```

```
    print("Thanks for watching!")
    print("-" * 88)


if __name__ == "__main__":
    do_scenario(boto3.resource("s3"))
```

- Untuk detail API, lihat topik berikut ini adalah Referensi API SDK untuk Python (Boto3)AWS .
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
require "aws-sdk-s3"

# Wraps the getting started scenario actions.
class ScenarioGettingStarted
  attr_reader :s3_resource

  # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
  def initialize(s3_resource)
    @s3_resource = s3_resource
```

```
  end

  # Creates a bucket with a random name in the currently configured account and
  # AWS Region.
  #
  # @return [Aws::S3::Bucket] The newly created bucket.
  def create_bucket
    bucket = @s3_resource.create_bucket(
      bucket: "doc-example-bucket-#{Random.uuid}",
      create_bucket_configuration: {
        location_constraint: "us-east-1" # Note: only certain regions permitted
      }
    )
    puts("Created demo bucket named #{bucket.name}.")
  rescue Aws::Errors::ServiceError => e
    puts("Tried and failed to create demo bucket.")
    puts("\t#{e.code}: #{e.message}")
    puts("\nCan't continue the demo without a bucket!")
    raise
  else
    bucket
  end

  # Requests a file name from the user.
  #
  # @return The name of the file.
  def create_file
    File.open("demo.txt", w) { |f| f.write("This is a demo file.") }
  end

  # Uploads a file to an Amazon S3 bucket.
  #
  # @param bucket [Aws::S3::Bucket] The bucket object representing the upload
 destination
  # @return [Aws::S3::Object] The Amazon S3 object that contains the uploaded
 file.
  def upload_file(bucket)
    File.open("demo.txt", "w+") { |f| f.write("This is a demo file.") }
    s3_object = bucket.object(File.basename("demo.txt"))
    s3_object.upload_file("demo.txt")
    puts("Uploaded file demo.txt into bucket #{bucket.name} with key
 #{s3_object.key}.")
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't upload file demo.txt to #{bucket.name}.")
```

```
    puts("\t#{e.code}: #{e.message}")
    raise
  else
    s3_object
  end

  # Downloads an Amazon S3 object to a file.
  #
  # @param s3_object [Aws::S3::Object] The object to download.
  def download_file(s3_object)
    puts("\nDo you want to download #{s3_object.key} to a local file (y/n)? ")
    answer = gets.chomp.downcase
    if answer == "y"
      puts("Enter a name for the downloaded file: ")
      file_name = gets.chomp
      s3_object.download_file(file_name)
      puts("Object #{s3_object.key} successfully downloaded to #{file_name}.")
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't download #{s3_object.key}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  end

  # Copies an Amazon S3 object to a subfolder within the same bucket.
  #
  # @param source_object [Aws::S3::Object] The source object to copy.
  # @return [Aws::S3::Object, nil] The destination object.
  def copy_object(source_object)
    dest_object = nil
    puts("\nDo you want to copy #{source_object.key} to a subfolder in your
 bucket (y/n)? ")
    answer = gets.chomp.downcase
    if answer == "y"
      dest_object = source_object.bucket.object("demo-folder/
#{source_object.key}")
      dest_object.copy_from(source_object)
      puts("Copied #{source_object.key} to #{dest_object.key}.")
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't copy #{source_object.key}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  else
```

```
    dest_object
  end

  # Lists the objects in an Amazon S3 bucket.
  #
  # @param bucket [Aws::S3::Bucket] The bucket to query.
  def list_objects(bucket)
    puts("\nYour bucket contains the following objects:")
    bucket.objects.each do |obj|
      puts("\t#{obj.key}")
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't list the objects in bucket #{bucket.name}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  end

  # Deletes the objects in an Amazon S3 bucket and deletes the bucket.
  #
  # @param bucket [Aws::S3::Bucket] The bucket to empty and delete.
  def delete_bucket(bucket)
    puts("\nDo you want to delete all of the objects as well as the bucket (y/n)?
 ")
    answer = gets.chomp.downcase
    if answer == "y"
      bucket.objects.batch_delete!
      bucket.delete
      puts("Emptied and deleted bucket #{bucket.name}.\n")
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't empty and delete bucket #{bucket.name}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  end
end

# Runs the Amazon S3 getting started scenario.
def run_scenario(scenario)
  puts("-" * 88)
  puts("Welcome to the Amazon S3 getting started demo!")
  puts("-" * 88)

  bucket = scenario.create_bucket
  s3_object = scenario.upload_file(bucket)
```

```
  scenario.download_file(s3_object)
  scenario.copy_object(s3_object)
  scenario.list_objects(bucket)
  scenario.delete_bucket(bucket)

  puts("Thanks for watching!")
  puts("-" * 88)
rescue Aws::Errors::ServiceError
  puts("Something went wrong with the demo!")
end

run_scenario(ScenarioGettingStarted.new(Aws::S3::Resource.new)) if $PROGRAM_NAME
 == __FILE__
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Ruby .
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Kode untuk peti biner yang menjalankan skenario.

```
use aws_config::meta::region::RegionProviderChain;
```

```
use aws_sdk_s3::{config::Region, Client};
use s3_service::error::Error;
use uuid::Uuid;


#[tokio::main]
async fn main() -> Result<(), Error> {
    let (region, client, bucket_name, file_name, key, target_key) =
 initialize_variables().await;

    if let Err(e) = run_s3_operations(region, client, bucket_name, file_name,
 key, target_key).await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (Region, Client, String, String, String,
 String) {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));
    let region = region_provider.region().await.unwrap();

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = Client::new(&shared_config);

    let bucket_name = format!("doc-example-bucket-{}", Uuid::new_v4());

    let file_name = "s3/testfile.txt".to_string();
    let key = "test file key name".to_string();
    let target_key = "target_key".to_string();

    (region, client, bucket_name, file_name, key, target_key)
}

async fn run_s3_operations(
    region: Region,
    client: Client,
    bucket_name: String,
    file_name: String,
    key: String,
```

```
    target_key: String,
) -> Result<(), Error> {
    s3_service::create_bucket(&client, &bucket_name, region.as_ref()).await?;
    s3_service::upload_object(&client, &bucket_name, &file_name, &key).await?;
    let _object = s3_service::download_object(&client, &bucket_name, &key).await;
    s3_service::copy_object(&client, &bucket_name, &key, &target_key).await?;
    s3_service::list_objects(&client, &bucket_name).await?;
    s3_service::delete_objects(&client, &bucket_name).await?;
    s3_service::delete_bucket(&client, &bucket_name).await?;

    Ok(())
}
```

Peti pustaka dengan tindakan umum yang dipanggil oleh biner.

```
use aws_sdk_s3::operation::{
    copy_object::{CopyObjectError, CopyObjectOutput},
    create_bucket::{CreateBucketError, CreateBucketOutput},
    get_object::{GetObjectError, GetObjectOutput},
    list_objects_v2::ListObjectsV2Output,
    put_object::{PutObjectError, PutObjectOutput},
};
use aws_sdk_s3::types::{
    BucketLocationConstraint, CreateBucketConfiguration, Delete,
 ObjectIdentifier,
};
use aws_sdk_s3::{error::SdkError, primitives::ByteStream, Client};
use error::Error;
use std::path::Path;
use std::str;

pub mod error;

pub async fn delete_bucket(client: &Client, bucket_name: &str) -> Result<(),
 Error> {
    client.delete_bucket().bucket(bucket_name).send().await?;
    println!("Bucket deleted");
    Ok(())
}
```

```
pub async fn delete_objects(client: &Client, bucket_name: &str) ->
 Result<Vec<String>, Error> {
    let objects = client.list_objects_v2().bucket(bucket_name).send().await?;

    let mut delete_objects: Vec<ObjectIdentifier> = vec![];
    for obj in objects.contents() {
        let obj_id = ObjectIdentifier::builder()
            .set_key(Some(obj.key().unwrap().to_string()))
            .build()
            .map_err(Error::from)?;
        delete_objects.push(obj_id);
    }

    let return_keys = delete_objects.iter().map(|o| o.key.clone()).collect();

    if !delete_objects.is_empty() {
        client
            .delete_objects()
            .bucket(bucket_name)
            .delete(
                Delete::builder()
                    .set_objects(Some(delete_objects))
                    .build()
                    .map_err(Error::from)?,
            )
            .send()
            .await?;
    }

    let objects: ListObjectsV2Output =
 client.list_objects_v2().bucket(bucket_name).send().await?;

    eprintln!("{objects:?}");

    match objects.key_count {
        Some(0) => Ok(return_keys),
        _ => Err(Error::unhandled(
            "There were still objects left in the bucket.",
        )),
    }
}

pub async fn list_objects(client: &Client, bucket: &str) -> Result<(), Error> {
    let mut response = client
```

```
        .list_objects_v2()
        .bucket(bucket.to_owned())
        .max_keys(10) // In this example, go 10 at a time.
        .into_paginator()
        .send();

    while let Some(result) = response.next().await {
        match result {
            Ok(output) => {
                for object in output.contents() {
                    println!(" - {}", object.key().unwrap_or("Unknown"));
                }
            }
            Err(err) => {
                eprintln!("{err:?}")
            }
        }
    }

    Ok(())
}

pub async fn copy_object(
    client: &Client,
    bucket_name: &str,
    object_key: &str,
    target_key: &str,
) -> Result<CopyObjectOutput, SdkError<CopyObjectError>> {
    let mut source_bucket_and_object: String = "".to_owned();
    source_bucket_and_object.push_str(bucket_name);
    source_bucket_and_object.push('/');
    source_bucket_and_object.push_str(object_key);

    client
        .copy_object()
        .copy_source(source_bucket_and_object)
        .bucket(bucket_name)
        .key(target_key)
        .send()
        .await
}

pub async fn download_object(
    client: &Client,
```

```
    bucket_name: &str,
    key: &str,
) -> Result<GetObjectOutput, SdkError<GetObjectError>> {
    client
        .get_object()
        .bucket(bucket_name)
        .key(key)
        .send()
        .await
}

pub async fn upload_object(
    client: &Client,
    bucket_name: &str,
    file_name: &str,
    key: &str,
) -> Result<PutObjectOutput, SdkError<PutObjectError>> {
    let body = ByteStream::from_path(Path::new(file_name)).await;
    client
        .put_object()
        .bucket(bucket_name)
        .key(key)
        .body(body.unwrap())
        .send()
        .await
}

pub async fn create_bucket(
    client: &Client,
    bucket_name: &str,
    region: &str,
) -> Result<CreateBucketOutput, SdkError<CreateBucketError>> {
    let constraint = BucketLocationConstraint::from(region);
    let cfg = CreateBucketConfiguration::builder()
        .location_constraint(constraint)
        .build();
    client
        .create_bucket()
        .create_bucket_configuration(cfg)
        .bucket(bucket_name)
        .send()
        .await
}
```

- Untuk detail API, lihat topik berikut dalam referensi API SDK untuk Rust AWS .
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

SAP ABAP

SDK untuk SAP ABAP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
DATA(lo_session) = /aws1/cl_rt_session_aws=>create( cv_pfl ).
DATA(lo_s3) = /aws1/cl_s3_factory=>create( lo_session ).

" Create an Amazon Simple Storage Service (Amazon S3) bucket. "
TRY.
    lo_s3->createbucket(
        iv_bucket = iv_bucket_name
    ).
    MESSAGE 'S3 bucket created.' TYPE 'I'.
  CATCH /aws1/cx_s3_bucketalrdyexists.
    MESSAGE 'Bucket name already exists.' TYPE 'E'.
  CATCH /aws1/cx_s3_bktalrdyownedbyyou.
    MESSAGE 'Bucket already exists and is owned by you.' TYPE 'E'.
ENDTRY.


"Upload an object to an S3 bucket."
```

```
    TRY.
        "Get contents of file from application server."
        DATA lv_file_content TYPE xstring.
        OPEN DATASET iv_key FOR INPUT IN BINARY MODE.
        READ DATASET iv_key INTO lv_file_content.
        CLOSE DATASET iv_key.

        lo_s3->putobject(
            iv_bucket = iv_bucket_name
            iv_key = iv_key
            iv_body = lv_file_content
        ).
        MESSAGE 'Object uploaded to S3 bucket.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
    ENDTRY.

    " Get an object from a bucket. "
    TRY.
        DATA(lo_result) = lo_s3->getobject(
                    iv_bucket = iv_bucket_name
                    iv_key = iv_key
                ).
        DATA(lv_object_data) = lo_result->get_body( ).
        MESSAGE 'Object retrieved from S3 bucket.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
      CATCH /aws1/cx_s3_nosuchkey.
        MESSAGE 'Object key does not exist.' TYPE 'E'.
    ENDTRY.

    " Copy an object to a subfolder in a bucket. "
    TRY.
        lo_s3->copyobject(
          iv_bucket = iv_bucket_name
          iv_key = |{ iv_copy_to_folder }/{ iv_key }|
          iv_copysource = |{ iv_bucket_name }/{ iv_key }|
        ).
        MESSAGE 'Object copied to a subfolder.' TYPE 'I'.
      CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
      CATCH /aws1/cx_s3_nosuchkey.
        MESSAGE 'Object key does not exist.' TYPE 'E'.
    ENDTRY.
```

```
" List objects in the bucket. "
TRY.
    DATA(lo_list) = lo_s3->listobjects(
       iv_bucket = iv_bucket_name
     ).
    MESSAGE 'Retrieved list of objects in S3 bucket.' TYPE 'I'.
  CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.
DATA text TYPE string VALUE 'Object List - '.
DATA lv_object_key TYPE /aws1/s3_objectkey.
LOOP AT lo_list->get_contents( ) INTO DATA(lo_object).
  lv_object_key = lo_object->get_key( ).
  CONCATENATE lv_object_key ', ' INTO text.
ENDLOOP.
MESSAGE text TYPE'I'.

" Delete the objects in a bucket. "
TRY.
    lo_s3->deleteobject(
        iv_bucket = iv_bucket_name
        iv_key = iv_key
    ).
    lo_s3->deleteobject(
        iv_bucket = iv_bucket_name
        iv_key = |{ iv_copy_to_folder }/{ iv_key }|
    ).
    MESSAGE 'Objects deleted from S3 bucket.' TYPE 'I'.
  CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.


" Delete the bucket. "
TRY.
    lo_s3->deletebucket(
        iv_bucket = iv_bucket_name
    ).
    MESSAGE 'Deleted S3 bucket.' TYPE 'I'.
  CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.
```

- Untuk detail API, lihat topik berikut di referensi API SDK untuk SAP ABAP AWS .
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Swift

SDK untuk Swift

> **Note**
>
> Ini adalah dokumentasi prarilis untuk SDK dalam rilis pratinjau. Dokumentasi ini dapat berubah.

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Kelas Swift yang menangani panggilan ke SDK untuk Swift.

```
import Foundation
import AWSS3
import ClientRuntime
import AWSClientRuntime

/// A class containing all the code that interacts with the AWS SDK for Swift.
public class ServiceHandler {
    let client: S3Client
```

```
    /// Initialize and return a new ``ServiceHandler`` object, which is used to
drive the AWS calls
    /// used for the example.
    ///
    /// - Returns: A new ``ServiceHandler`` object, ready to be called to
    ///            execute AWS operations.
    public init() async {
        do {
            client = try S3Client(region: "us-east-2")
        } catch {
            print("ERROR: ", dump(error, name: "Initializing S3 client"))
            exit(1)
        }
    }

    /// Create a new user given the specified name.
    ///
    /// - Parameters:
    ///   - name: Name of the bucket to create.
    /// Throws an exception if an error occurs.
    public func createBucket(name: String) async throws {
        let config = S3ClientTypes.CreateBucketConfiguration(
            locationConstraint: .usEast2
        )
        let input = CreateBucketInput(
            bucket: name,
            createBucketConfiguration: config
        )
        _ = try await client.createBucket(input: input)
    }

    /// Delete a bucket.
    /// - Parameter name: Name of the bucket to delete.
    public func deleteBucket(name: String) async throws {
        let input = DeleteBucketInput(
            bucket: name
        )
        _ = try await client.deleteBucket(input: input)
    }

    /// Upload a file from local storage to the bucket.
    /// - Parameters:
    ///   - bucket: Name of the bucket to upload the file to.
    ///   - key: Name of the file to create.
```

```
    ///   - file: Path name of the file to upload.
    public func uploadFile(bucket: String, key: String, file: String) async
throws {
        let fileUrl = URL(fileURLWithPath: file)
        let fileData = try Data(contentsOf: fileUrl)
        let dataStream = ByteStream.from(data: fileData)

        let input = PutObjectInput(
            body: dataStream,
            bucket: bucket,
            key: key
        )
        _ = try await client.putObject(input: input)
    }

    /// Create a file in the specified bucket with the given name. The new
    /// file's contents are uploaded from a `Data` object.
    ///
    /// - Parameters:
    ///   - bucket: Name of the bucket to create a file in.
    ///   - key: Name of the file to create.
    ///   - data: A `Data` object to write into the new file.
    public func createFile(bucket: String, key: String, withData data: Data)
async throws {
        let dataStream = ByteStream.from(data: data)

        let input = PutObjectInput(
            body: dataStream,
            bucket: bucket,
            key: key
        )
        _ = try await client.putObject(input: input)
    }

    /// Download the named file to the given directory on the local device.
    ///
    /// - Parameters:
    ///   - bucket: Name of the bucket that contains the file to be copied.
    ///   - key: The name of the file to copy from the bucket.
    ///   - to: The path of the directory on the local device where you want to
    ///     download the file.
    public func downloadFile(bucket: String, key: String, to: String) async
throws {
        let fileUrl = URL(fileURLWithPath: to).appendingPathComponent(key)
```

```
        let input = GetObjectInput(
            bucket: bucket,
            key: key
        )
        let output = try await client.getObject(input: input)

        // Get the data stream object. Return immediately if there isn't one.
        guard let body = output.body,
              let data = try await body.readData() else {
            return
        }
        try data.write(to: fileUrl)
    }

    /// Read the specified file from the given S3 bucket into a Swift
    /// `Data` object.
    ///
    /// - Parameters:
    ///   - bucket: Name of the bucket containing the file to read.
    ///   - key: Name of the file within the bucket to read.
    ///
    /// - Returns: A `Data` object containing the complete file data.
    public func readFile(bucket: String, key: String) async throws -> Data {
        let input = GetObjectInput(
            bucket: bucket,
            key: key
        )
        let output = try await client.getObject(input: input)

        // Get the stream and return its contents in a `Data` object. If
        // there is no stream, return an empty `Data` object instead.
        guard let body = output.body,
              let data = try await body.readData() else {
            return "".data(using: .utf8)!
        }

        return data
    }

    /// Copy a file from one bucket to another.
    ///
    /// - Parameters:
    ///   - sourceBucket: Name of the bucket containing the source file.
```

```
    ///   - name: Name of the source file.
    ///   - destBucket: Name of the bucket to copy the file into.
    public func copyFile(from sourceBucket: String, name: String, to destBucket:
 String) async throws {
        let srcUrl = ("\(sourceBucket)/
\(name)").addingPercentEncoding(withAllowedCharacters: .urlPathAllowed)

        let input = CopyObjectInput(
            bucket: destBucket,
            copySource: srcUrl,
            key: name
        )
        _ = try await client.copyObject(input: input)
    }

    /// Deletes the specified file from Amazon S3.
    ///
    /// - Parameters:
    ///   - bucket: Name of the bucket containing the file to delete.
    ///   - key: Name of the file to delete.
    ///
    public func deleteFile(bucket: String, key: String) async throws {
        let input = DeleteObjectInput(
            bucket: bucket,
            key: key
        )

        do {
            _ = try await client.deleteObject(input: input)
        } catch {
            throw error
        }
    }

    /// Returns an array of strings, each naming one file in the
    /// specified bucket.
    ///
    /// - Parameter bucket: Name of the bucket to get a file listing for.
    /// - Returns: An array of `String` objects, each giving the name of
    ///            one file contained in the bucket.
    public func listBucketFiles(bucket: String) async throws -> [String] {
        let input = ListObjectsV2Input(
            bucket: bucket
        )
```

```
        let output = try await client.listObjectsV2(input: input)
        var names: [String] = []

        guard let objList = output.contents else {
            return []
        }

        for obj in objList {
            if let objName = obj.key {
                names.append(objName)
            }
        }

        return names
    }
}
```

Program baris perintah Swift untuk mengelola panggilan SDK.

```
import Foundation
import ServiceHandler
import ArgumentParser

/// The command-line arguments and options available for this
/// example command.
struct ExampleCommand: ParsableCommand {
    @Argument(help: "Name of the S3 bucket to create")
    var bucketName: String

    @Argument(help: "Pathname of the file to upload to the S3 bucket")
    var uploadSource: String

    @Argument(help: "The name (key) to give the file in the S3 bucket")
    var objName: String

    @Argument(help: "S3 bucket to copy the object to")
    var destBucket: String

    @Argument(help: "Directory where you want to download the file from the S3
 bucket")
    var downloadDir: String
```

```
    static var configuration = CommandConfiguration(
        commandName: "s3-basics",
        abstract: "Demonstrates a series of basic AWS S3 functions.",
        discussion: """
        Performs the following Amazon S3 commands:

        * `CreateBucket`
        * `PutObject`
        * `GetObject`
        * `CopyObject`
        * `ListObjects`
        * `DeleteObjects`
        * `DeleteBucket`
        """
    )

    /// Called by ``main()`` to do the actual running of the AWS
    /// example.
    func runAsync() async throws {
        let serviceHandler = await ServiceHandler()

        // 1. Create the bucket.
        print("Creating the bucket \(bucketName)...")
        try await serviceHandler.createBucket(name: bucketName)

        // 2. Upload a file to the bucket.
        print("Uploading the file \(uploadSource)...")
        try await serviceHandler.uploadFile(bucket: bucketName, key: objName,
 file: uploadSource)

        // 3. Download the file.
        print("Downloading the file \(objName) to \(downloadDir)...")
        try await serviceHandler.downloadFile(bucket: bucketName, key: objName,
 to: downloadDir)

        // 4. Copy the file to another bucket.
        print("Copying the file to the bucket \(destBucket)...")
        try await serviceHandler.copyFile(from: bucketName, name: objName, to:
 destBucket)

        // 5. List the contents of the bucket.

        print("Getting a list of the files in the bucket \(bucketName)")
```

```
        let fileList = try await serviceHandler.listBucketFiles(bucket:
 bucketName)
        let numFiles = fileList.count
        if numFiles != 0 {
            print("\(numFiles) file\((numFiles > 1) ? "s" : "") in bucket
 \(bucketName):")
            for name in fileList {
                print("  \(name)")
            }
        } else {
            print("No files found in bucket \(bucketName)")
        }

        // 6. Delete the objects from the bucket.

        print("Deleting the file \(objName) from the bucket \(bucketName)...")
        try await serviceHandler.deleteFile(bucket: bucketName, key: objName)
        print("Deleting the file \(objName) from the bucket \(destBucket)...")
        try await serviceHandler.deleteFile(bucket: destBucket, key: objName)

        // 7. Delete the bucket.
        print("Deleting the bucket \(bucketName)...")
        try await serviceHandler.deleteBucket(name: bucketName)

        print("Done.")
    }
}

//
// Main program entry point.
//
@main
struct Main {
    static func main() async {
        let args = Array(CommandLine.arguments.dropFirst())

        do {
            let command = try ExampleCommand.parse(args)
            try await command.runAsync()
        } catch {
            ExampleCommand.exit(withError: error)
        }
    }
}
```

- Untuk detail API, lihat topik berikut di Referensi API SDK untuk Swift AWS .
  - CopyObject
  - CreateBucket
  - DeleteBucket
  - DeleteObjects
  - GetObject
  - ListObjectsV2
  - PutObject

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai enkripsi untuk objek Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara memulai enkripsi untuk objek Amazon S3.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
using System;
using System.IO;
using System.Security.Cryptography;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to apply client encryption to an object in an
```

```
    /// Amazon Simple Storage Service (Amazon S3) bucket.
    /// </summary>
    public class SSEClientEncryption
    {
        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";
            string keyName = "exampleobject.txt";
            string copyTargetKeyName = "examplecopy.txt";

            // If the AWS Region defined for your default user is different
            // from the Region where your Amazon S3 bucket is located,
            // pass the Region name to the Amazon S3 client object's constructor.
            // For example: RegionEndpoint.USWest2.
            IAmazonS3 client = new AmazonS3Client();

            try
            {
                // Create an encryption key.
                Aes aesEncryption = Aes.Create();
                aesEncryption.KeySize = 256;
                aesEncryption.GenerateKey();
                string base64Key = Convert.ToBase64String(aesEncryption.Key);

                // Upload the object.
                PutObjectRequest putObjectRequest = await
 UploadObjectAsync(client, bucketName, keyName, base64Key);

                // Download the object and verify that its contents match what
 you uploaded.
                await DownloadObjectAsync(client, bucketName, keyName, base64Key,
 putObjectRequest);

                // Get object metadata and verify that the object uses AES-256
 encryption.
                await GetObjectMetadataAsync(client, bucketName, keyName,
 base64Key);

                // Copy both the source and target objects using server-side
 encryption with
                // an encryption key.
                await CopyObjectAsync(client, bucketName, keyName,
 copyTargetKeyName, aesEncryption, base64Key);
            }
```

```
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error: {ex.Message}");
            }
        }

        /// <summary>
        /// Uploads an object to an Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// PutObjectAsync.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket to which
the
        /// object will be uploaded.</param>
        /// <param name="keyName">The name of the object to upload to the Amazon
S3
        /// bucket.</param>
        /// <param name="base64Key">The encryption key.</param>
        /// <returns>The PutObjectRequest object for use by
DownloadObjectAsync.</returns>
        public static async Task<PutObjectRequest> UploadObjectAsync(
            IAmazonS3 client,
            string bucketName,
            string keyName,
            string base64Key)
        {
            PutObjectRequest putObjectRequest = new PutObjectRequest
            {
                BucketName = bucketName,
                Key = keyName,
                ContentBody = "sample text",
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key,
            };
            PutObjectResponse putObjectResponse = await
client.PutObjectAsync(putObjectRequest);
            return putObjectRequest;
        }

        /// <summary>
        /// Downloads an encrypted object from an Amazon S3 bucket.
        /// </summary>
```

```
        /// <param name="client">The initialized Amazon S3 client object used to
 call
        /// GetObjectAsync.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket where the
 object
        /// is located.</param>
        /// <param name="keyName">The name of the Amazon S3 object to download.</
param>
        /// <param name="base64Key">The encryption key used to encrypt the
        /// object.</param>
        /// <param name="putObjectRequest">The PutObjectRequest used to upload
        /// the object.</param>
        public static async Task DownloadObjectAsync(
            IAmazonS3 client,
            string bucketName,
            string keyName,
            string base64Key,
            PutObjectRequest putObjectRequest)
        {
            GetObjectRequest getObjectRequest = new GetObjectRequest
            {
                BucketName = bucketName,
                Key = keyName,

                // Provide encryption information for the object stored in Amazon
 S3.
                ServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key,
            };

            using (GetObjectResponse getResponse = await
 client.GetObjectAsync(getObjectRequest))
            using (StreamReader reader = new
 StreamReader(getResponse.ResponseStream))
            {
                string content = reader.ReadToEnd();
                if (string.Compare(putObjectRequest.ContentBody, content) == 0)
                {
                    Console.WriteLine("Object content is same as we uploaded");
                }
                else
                {
                    Console.WriteLine("Error...Object content is not same.");
```

```
                }

                if (getResponse.ServerSideEncryptionCustomerMethod ==
ServerSideEncryptionCustomerMethod.AES256)
                {
                    Console.WriteLine("Object encryption method is AES256, same
as we set");
                }
                else
                {
                    Console.WriteLine("Error...Object encryption method is not
the same as AES256 we set");
                }
            }
        }

        /// <summary>
        /// Retrieves the metadata associated with an Amazon S3 object.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used
        /// to call GetObjectMetadataAsync.</param>
        /// <param name="bucketName">The name of the Amazon S3 bucket containing
the
        /// object for which we want to retrieve metadata.</param>
        /// <param name="keyName">The name of the object for which we wish to
        /// retrieve the metadata.</param>
        /// <param name="base64Key">The encryption key associated with the
        /// object.</param>
        public static async Task GetObjectMetadataAsync(
            IAmazonS3 client,
            string bucketName,
            string keyName,
            string base64Key)
        {
            GetObjectMetadataRequest getObjectMetadataRequest = new
GetObjectMetadataRequest
            {
                BucketName = bucketName,
                Key = keyName,

                // The object stored in Amazon S3 is encrypted, so provide the
necessary encryption information.
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
```

```
                ServerSideEncryptionCustomerProvidedKey = base64Key,
            };

            GetObjectMetadataResponse getObjectMetadataResponse = await
client.GetObjectMetadataAsync(getObjectMetadataRequest);
            Console.WriteLine("The object metadata show encryption method used
is: {0}", getObjectMetadataResponse.ServerSideEncryptionCustomerMethod);
        }

        /// <summary>
        /// Copies an encrypted object from one Amazon S3 bucket to another.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// CopyObjectAsync.</param>
        /// <param name="bucketName">The Amazon S3 bucket containing the object
        /// to copy.</param>
        /// <param name="keyName">The name of the object to copy.</param>
        /// <param name="copyTargetKeyName">The Amazon S3 bucket to which the
object
        /// will be copied.</param>
        /// <param name="aesEncryption">The encryption type to use.</param>
        /// <param name="base64Key">The encryption key to use.</param>
        public static async Task CopyObjectAsync(
            IAmazonS3 client,
            string bucketName,
            string keyName,
            string copyTargetKeyName,
            Aes aesEncryption,
            string base64Key)
        {
            aesEncryption.GenerateKey();
            string copyBase64Key = Convert.ToBase64String(aesEncryption.Key);

            CopyObjectRequest copyRequest = new CopyObjectRequest
            {
                SourceBucket = bucketName,
                SourceKey = keyName,
                DestinationBucket = bucketName,
                DestinationKey = copyTargetKeyName,

                // Information about the source object's encryption.
                CopySourceServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
```

```
                CopySourceServerSideEncryptionCustomerProvidedKey = base64Key,

                // Information about the target object's encryption.
                ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = copyBase64Key,
            };
            await client.CopyObjectAsync(copyRequest);
        }
    }
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for .NET .
  - CopyObject
  - GetObject
  - GetObjectMetadata

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai tag untuk objek Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara memulai tag untuk objek Amazon S3.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Amazon;
```

```
    using Amazon.S3;
    using Amazon.S3.Model;

    /// <summary>
    /// This example shows how to work with tags in Amazon Simple Storage
    /// Service (Amazon S3) objects.
    /// </summary>
    public class ObjectTag
    {
        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";
            string keyName = "newobject.txt";
            string filePath = @"*** file path ***";

            // Specify your bucket region (an example region is shown).
            RegionEndpoint bucketRegion = RegionEndpoint.USWest2;

            var client = new AmazonS3Client(bucketRegion);
            await PutObjectsWithTagsAsync(client, bucketName, keyName, filePath);
        }

        /// <summary>
        /// This method uploads an object with tags. It then shows the tag
        /// values, changes the tags, and shows the new tags.
        /// </summary>
        /// <param name="client">The Initialized Amazon S3 client object used
        /// to call the methods to create and change an objects tags.</param>
        /// <param name="bucketName">A string representing the name of the
        /// bucket where the object will be stored.</param>
        /// <param name="keyName">A string representing the key name of the
        /// object to be tagged.</param>
        /// <param name="filePath">The directory location and file name of the
        /// object to be uploaded to the Amazon S3 bucket.</param>
        public static async Task PutObjectsWithTagsAsync(IAmazonS3 client, string
 bucketName, string keyName, string filePath)
        {
            try
            {
                // Create an object with tags.
                var putRequest = new PutObjectRequest
                {
                    BucketName = bucketName,
                    Key = keyName,
```

```
                    FilePath = filePath,
                    TagSet = new List<Tag>
                    {
                        new Tag { Key = "Keyx1", Value = "Value1" },
                        new Tag { Key = "Keyx2", Value = "Value2" },
                    },
                };

                PutObjectResponse response = await
client.PutObjectAsync(putRequest);

                // Now retrieve the new object's tags.
                GetObjectTaggingRequest getTagsRequest = new
GetObjectTaggingRequest()
                {
                    BucketName = bucketName,
                    Key = keyName,
                };

                GetObjectTaggingResponse objectTags = await
client.GetObjectTaggingAsync(getTagsRequest);

                // Display the tag values.
                objectTags.Tagging
                    .ForEach(t => Console.WriteLine($"Key: {t.Key}, Value:
{t.Value}"));

                Tagging newTagSet = new Tagging()
                {
                    TagSet = new List<Tag>
                    {
                        new Tag { Key = "Key3", Value = "Value3" },
                        new Tag { Key = "Key4", Value = "Value4" },
                    },
                };

                PutObjectTaggingRequest putObjTagsRequest = new
PutObjectTaggingRequest()
                {
                    BucketName = bucketName,
                    Key = keyName,
                    Tagging = newTagSet,
                };
```

```
                PutObjectTaggingResponse response2 = await
client.PutObjectTaggingAsync(putObjTagsRequest);

                // Retrieve the tags again and show the values.
                GetObjectTaggingRequest getTagsRequest2 = new
GetObjectTaggingRequest()
                {
                    BucketName = bucketName,
                    Key = keyName,
                };
                GetObjectTaggingResponse objectTags2 = await
client.GetObjectTaggingAsync(getTagsRequest2);

                objectTags2.Tagging
                    .ForEach(t => Console.WriteLine($"Key: {t.Key}, Value:
{t.Value}"));
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine(
                        $"Error: '{ex.Message}'");
            }
        }
    }
```

- Untuk detail API, lihat GetObjectTaggingdi Referensi AWS SDK for .NET API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Dapatkan konfigurasi penahanan hukum objek Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara mendapatkan konfigurasi penahanan legal bucket S3.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// Get the legal hold details for an S3 object.
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The object key.</param>
    /// <returns>The object legal hold details.</returns>
    public async Task<ObjectLockLegalHold> GetObjectLegalHold(string bucketName,
        string objectKey)
    {
        try
        {
            var request = new GetObjectLegalHoldRequest()
            {
                BucketName = bucketName,
                Key = objectKey
            };

            var response = await _amazonS3.GetObjectLegalHoldAsync(request);
            Console.WriteLine($"\tObject legal hold for {objectKey} in
{bucketName}: " +
                              $"\n\tStatus: {response.LegalHold.Status}");
            return response.LegalHold;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to fetch legal hold: '{ex.Message}'");
            return new ObjectLockLegalHold();
        }
    }
```

- Untuk detail API, lihat GetObjectLegalHolddi Referensi AWS SDK for .NET API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
   // Get the legal hold details for an S3 object.
   public ObjectLockLegalHold getObjectLegalHold(String bucketName, String
objectKey) {
       try {
           GetObjectLegalHoldRequest legalHoldRequest =
GetObjectLegalHoldRequest.builder()
               .bucket(bucketName)
               .key(objectKey)
               .build();

           GetObjectLegalHoldResponse response =
getClient().getObjectLegalHold(legalHoldRequest);
           System.out.println("Object legal hold for " + objectKey + " in " +
bucketName +
               ":\n\tStatus: " + response.legalHold().status());
           return response.legalHold();

       } catch (S3Exception ex) {
           System.out.println("\tUnable to fetch legal hold: '" +
ex.getMessage() + "'");
       }

       return null;
   }
```

- Untuk detail API, lihat GetObjectLegalHolddi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

import { fileURLToPath } from "url";
import { GetObjectLegalHoldCommand, S3Client } from "@aws-sdk/client-s3";

/**
 * @param {S3Client} client
 * @param {string} bucketName
 * @param {string} objectKey
 */
export const main = async (client, bucketName, objectKey) => {
  const command = new GetObjectLegalHoldCommand({
    Bucket: bucketName,
    Key: objectKey,
    // Optionally, you can provide additional parameters
    // ExpectedBucketOwner: "ACCOUNT_ID",
    // RequestPayer: "requester",
    // VersionId: "OBJECT_VERSION_ID",
  });

  try {
    const response = await client.send(command);
    console.log(`Legal Hold Status: ${response.LegalHold.Status}`);
  } catch (err) {
    console.error(err);
  }
};

// Invoke main function if this file was run directly.
if (process.argv[1] === fileURLToPath(import.meta.url)) {
  main(new S3Client(), "DOC-EXAMPLE-BUCKET", "OBJECT_KEY");
```

```
}
```

- Untuk detail API, lihat GetObjectLegalHolddi Referensi AWS SDK for JavaScript API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Bekerja dengan fitur kunci objek Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara bekerja dengan fitur kunci objek S3.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Jalankan skenario interaktif yang mendemonstrasikan fitur kunci objek Amazon S3.

```
using Amazon.S3;
using Amazon.S3.Model;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;
using Microsoft.Extensions.Logging;
using Microsoft.Extensions.Logging.Console;
using Microsoft.Extensions.Logging.Debug;

namespace S3ObjectLockScenario;

public static class S3ObjectLockWorkflow
{
    /*
```

```
   Before running this .NET code example, set up your development environment,
including your credentials.

   This .NET example performs the following tasks:
       1. Create test Amazon Simple Storage Service (S3) buckets with different
lock policies.
       2. Upload sample objects to each bucket.
       3. Set some Legal Hold and Retention Periods on objects and buckets.
       4. Investigate lock policies by viewing settings or attempting to delete
or overwrite objects.
       5. Clean up objects and buckets.
  */

   public static S3ActionsWrapper _s3ActionsWrapper = null!;
   public static IConfiguration _configuration = null!;
   private static string _resourcePrefix = null!;
   private static string noLockBucketName = null!;
   private static string lockEnabledBucketName = null!;
   private static string retentionAfterCreationBucketName = null!;
   private static List<string> bucketNames = new List<string>();
   private static List<string> fileNames = new List<string>();

   public static async Task Main(string[] args)
   {
       // Set up dependency injection for the Amazon service.
       using var host = Host.CreateDefaultBuilder(args)
           .ConfigureLogging(logging =>
               logging.AddFilter("System", LogLevel.Debug)
                   .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                   .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
           .ConfigureServices((_, services) =>
               services.AddAWSService<IAmazonS3>()
                   .AddTransient<S3ActionsWrapper>()
           )
           .Build();

       _configuration = new ConfigurationBuilder()
           .SetBasePath(Directory.GetCurrentDirectory())
           .AddJsonFile("settings.json") // Load settings from .json file.
           .AddJsonFile("settings.local.json",
               true) // Optionally, load local settings.
           .Build();
```

```
        ConfigurationSetup();

        ServicesSetup(host);

        try
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Welcome to the Amazon Simple Storage Service (S3)
 Object Locking Workflow Scenario.");
            Console.WriteLine(new string('-', 80));
            await Setup(true);

            await DemoActionChoices();

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Cleaning up resources.");
            Console.WriteLine(new string('-', 80));
            await Cleanup(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Amazon S3 Object Locking Workflow is complete.");
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine($"There was a problem: {ex.Message}");
            await Cleanup(true);
            Console.WriteLine(new string('-', 80));
        }
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _s3ActionsWrapper = host.Services.GetRequiredService<S3ActionsWrapper>();
    }

    /// <summary>
    /// Any setup operations needed.
```

```
    /// </summary>
    public static void ConfigurationSetup()
    {
        _resourcePrefix = _configuration["resourcePrefix"] ?? "dotnet-example";

        noLockBucketName = _resourcePrefix + "-no-lock";
        lockEnabledBucketName = _resourcePrefix + "-lock-enabled";
        retentionAfterCreationBucketName = _resourcePrefix + "-retention-after-
creation";

        bucketNames.Add(noLockBucketName);
        bucketNames.Add(lockEnabledBucketName);
        bucketNames.Add(retentionAfterCreationBucketName);
    }

    // <summary>
    /// Deploy necessary resources for the scenario.
    /// </summary>
    /// <param name="interactive">True to run as interactive.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> Setup(bool interactive)
    {
        Console.WriteLine(
            "\nFor this workflow, we will use the AWS SDK for .NET to create
 several S3\n" +
            "buckets and files to demonstrate working with S3 locking features.
\n");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you are ready to start.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("\nS3 buckets can be created either with or without
 object lock enabled.");
        await _s3ActionsWrapper.CreateBucketWithObjectLock(noLockBucketName,
 false);
        await _s3ActionsWrapper.CreateBucketWithObjectLock(lockEnabledBucketName,
 true);
        await
 _s3ActionsWrapper.CreateBucketWithObjectLock(retentionAfterCreationBucketName,
 false);

        Console.WriteLine("Press Enter to continue.");
```

```
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("\nA bucket can be configured to use object locking
with a default retention period.");
        await
_s3ActionsWrapper.ModifyBucketDefaultRetention(retentionAfterCreationBucketName,
true,
            ObjectLockRetentionMode.Governance, DateTime.UtcNow.AddDays(1));

        Console.WriteLine("Press Enter to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("\nObject lock policies can also be added to existing
buckets.");
        await _s3ActionsWrapper.EnableObjectLockOnBucket(lockEnabledBucketName);

        Console.WriteLine("Press Enter to continue.");
        if (interactive)
            Console.ReadLine();

        // Upload some files to the buckets.
        Console.WriteLine("\nNow let's add some test files:");
        var fileName = _configuration["exampleFileName"] ?? "exampleFile.txt";
        int fileCount = 2;
        // Create the file if it does not already exist.
        if (!File.Exists(fileName))
        {
            await using StreamWriter sw = File.CreateText(fileName);
            await sw.WriteLineAsync(
                "This is a sample file for uploading to a bucket.");
        }

        foreach (var bucketName in bucketNames)
        {
            for (int i = 0; i < fileCount; i++)
            {
                var numberedFileName = Path.GetFileNameWithoutExtension(fileName)
+ i + Path.GetExtension(fileName);
                fileNames.Add(numberedFileName);
                await _s3ActionsWrapper.UploadFileAsync(bucketName,
numberedFileName, fileName);
            }
```

```
        }
        Console.WriteLine("Press Enter to continue.");
        if (interactive)
            Console.ReadLine();

        if (!interactive)
            return true;
        Console.WriteLine("\nNow we can set some object lock policies on
individual files:");
        foreach (var bucketName in bucketNames)
        {
            for (int i = 0; i < fileNames.Count; i++)
            {
                // No modifications to the objects in the first bucket.
                if (bucketName != bucketNames[0])
                {
                    var exampleFileName = fileNames[i];
                    switch (i)
                    {
                        case 0:
                            {
                                var question =
                                    $"\nWould you like to add a legal hold to
{exampleFileName} in {bucketName}? (y/n)";
                                if (GetYesNoResponse(question))
                                {
                                    // Set a legal hold.
                                    await
_s3ActionsWrapper.ModifyObjectLegalHold(bucketName, exampleFileName,
ObjectLockLegalHoldStatus.On);

                                }
                                break;
                            }
                        case 1:
                            {
                                var question =
                                    $"\nWould you like to add a 1 day Governance
retention period to {exampleFileName} in {bucketName}? (y/n)" +
                                    "\nReminder: Only a user with the
s3:BypassGovernanceRetention permission will be able to delete this file or its
bucket until the retention period has expired.";
                                if (GetYesNoResponse(question))
                                {
```

```
                                    // Set a Governance mode retention period for
1 day.
                                    await
_s3ActionsWrapper.ModifyObjectRetentionPeriod(
                                        bucketName, exampleFileName,
                                        ObjectLockRetentionMode.Governance,
                                        DateTime.UtcNow.AddDays(1));
                                }
                                break;
                            }
                        }
                    }
                }
            }
            Console.WriteLine(new string('-', 80));
            return true;
        }

        // <summary>
        /// List all of the current buckets and objects.
        /// </summary>
        /// <param name="interactive">True to run as interactive.</param>
        /// <returns>The list of buckets and objects.</returns>
        public static async Task<List<S3ObjectVersion>> ListBucketsAndObjects(bool
interactive)
        {
            var allObjects = new List<S3ObjectVersion>();
            foreach (var bucketName in bucketNames)
            {
                var objectsInBucket = await
_s3ActionsWrapper.ListBucketObjectsAndVersions(bucketName);
                foreach (var objectKey in objectsInBucket.Versions)
                {
                    allObjects.Add(objectKey);
                }
            }

            if (interactive)
            {
                Console.WriteLine("\nCurrent buckets and objects:\n");
                int i = 0;
                foreach (var bucketObject in allObjects)
                {
                    i++;
```

```
                Console.WriteLine(
                    $"{i}: {bucketObject.Key} \n\tBucket:
{bucketObject.BucketName}\n\tVersion: {bucketObject.VersionId}");
            }
        }

        return allObjects;
    }

    /// <summary>
    /// Present the user with the demo action choices.
    /// </summary>
    /// <returns>Async task.</returns>
    public static async Task<bool> DemoActionChoices()
    {
        var choices = new string[]{
            "List all files in buckets.",
            "Attempt to delete a file.",
            "Attempt to delete a file with retention period bypass.",
            "Attempt to overwrite a file.",
            "View the object and bucket retention settings for a file.",
            "View the legal hold settings for a file.",
            "Finish the workflow."};

        var choice = 0;
        // Keep asking the user until they choose to move on.
        while (choice != 6)
        {
            Console.WriteLine(new string('-', 80));
            choice = GetChoiceResponse(
                "\nExplore the S3 locking features by selecting one of the
following choices:"
                , choices);
            Console.WriteLine(new string('-', 80));
            switch (choice)
            {
                case 0:
                    {
                        await ListBucketsAndObjects(true);
                        break;
                    }
                case 1:
                    {
```

```
                        Console.WriteLine("\nEnter the number of the object to
delete:");
                        var allFiles = await ListBucketsAndObjects(true);
                        var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
                        await
_s3ActionsWrapper.DeleteObjectFromBucket(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key, false, allFiles[fileChoice].VersionId);
                        break;
                    }
                case 2:
                    {
                        Console.WriteLine("\nEnter the number of the object to
delete:");
                        var allFiles = await ListBucketsAndObjects(true);
                        var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
                        await
_s3ActionsWrapper.DeleteObjectFromBucket(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key, true, allFiles[fileChoice].VersionId);
                        break;
                    }
                case 3:
                    {
                        var allFiles = await ListBucketsAndObjects(true);
                        Console.WriteLine("\nEnter the number of the object to
overwrite:");
                        var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
                        // Create the file if it does not already exist.
                        if (!File.Exists(allFiles[fileChoice].Key))
                        {
                            await using StreamWriter sw =
File.CreateText(allFiles[fileChoice].Key);
                            await sw.WriteLineAsync(
                                "This is a sample file for uploading to a
bucket.");
                        }
                        await
_s3ActionsWrapper.UploadFileAsync(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key, allFiles[fileChoice].Key);
                        break;
                    }
                case 4:
```

```
                    {
                        var allFiles = await ListBucketsAndObjects(true);
                        Console.WriteLine("\nEnter the number of the object and
bucket to view:");
                        var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
                        await
_s3ActionsWrapper.GetObjectRetention(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key);
                        await
_s3ActionsWrapper.GetBucketObjectLockConfiguration(allFiles[fileChoice].BucketName);
                        break;
                    }
                case 5:
                    {
                        var allFiles = await ListBucketsAndObjects(true);
                        Console.WriteLine("\nEnter the number of the object to
view:");
                        var fileChoice = GetChoiceResponse(null,
allFiles.Select(f => f.Key).ToArray());
                        await
_s3ActionsWrapper.GetObjectLegalHold(allFiles[fileChoice].BucketName,
allFiles[fileChoice].Key);
                        break;
                    }
            }
        }
        return true;
    }

    // <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to run as interactive.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> Cleanup(bool interactive)
    {
        Console.WriteLine(new string('-', 80));

        if (!interactive || GetYesNoResponse("Do you want to clean up all files
and buckets? (y/n) "))
        {
            // Remove all locks and delete all buckets and objects.
            var allFiles = await ListBucketsAndObjects(false);
```

```
            foreach (var fileInfo in allFiles)
            {
                // Check for a legal hold.
                var legalHold = await
_s3ActionsWrapper.GetObjectLegalHold(fileInfo.BucketName, fileInfo.Key);
                if (legalHold?.Status?.Value == ObjectLockLegalHoldStatus.On)
                {
                    await
_s3ActionsWrapper.ModifyObjectLegalHold(fileInfo.BucketName, fileInfo.Key,
ObjectLockLegalHoldStatus.Off);
                }

                // Check for a retention period.
                var retention = await
_s3ActionsWrapper.GetObjectRetention(fileInfo.BucketName, fileInfo.Key);
                var hasRetentionPeriod = retention?.Mode ==
ObjectLockRetentionMode.Governance && retention.RetainUntilDate >
DateTime.UtcNow.Date;
                await
_s3ActionsWrapper.DeleteObjectFromBucket(fileInfo.BucketName, fileInfo.Key,
hasRetentionPeriod, fileInfo.VersionId);
            }

            foreach (var bucketName in bucketNames)
            {
                await _s3ActionsWrapper.DeleteBucketByName(bucketName);
            }

        }
        else
        {
            Console.WriteLine(
                "Ok, we'll leave the resources intact.\n" +
                "Don't forget to delete them when you're done with them or you
might incur unexpected charges."
            );
        }

        Console.WriteLine(new string('-', 80));
        return true;
    }

    /// <summary>
    /// Helper method to get a yes or no response from the user.
```

```
    /// </summary>
    /// <param name="question">The question string to print on the console.</
param>
    /// <returns>True if the user responds with a yes.</returns>
    private static bool GetYesNoResponse(string question)
    {
        Console.WriteLine(question);
        var ynResponse = Console.ReadLine();
        var response = ynResponse != null && ynResponse.Equals("y",
 StringComparison.InvariantCultureIgnoreCase);
        return response;
    }

    /// <summary>
    /// Helper method to get a choice response from the user.
    /// </summary>
    /// <param name="question">The question string to print on the console.</
param>
    /// <param name="choices">The choices to print on the console.</param>
    /// <returns>The index of the selected choice</returns>
    private static int GetChoiceResponse(string? question, string[] choices)
    {
        if (question != null)
        {
            Console.WriteLine(question);

            for (int i = 0; i < choices.Length; i++)
            {
                Console.WriteLine($"\t{i + 1}. {choices[i]}");
            }
        }

        var choiceNumber = 0;
        while (choiceNumber < 1 || choiceNumber > choices.Length)
        {
            var choice = Console.ReadLine();
            Int32.TryParse(choice, out choiceNumber);
        }

        return choiceNumber - 1;
    }
}
```

Kelas pembungkus untuk fungsi S3.

```
using System.Net;
using Amazon.S3;
using Amazon.S3.Model;
using Microsoft.Extensions.Configuration;

namespace S3ObjectLockScenario;

/// <summary>
/// Encapsulate the Amazon S3 operations.
/// </summary>
public class S3ActionsWrapper
{
    private readonly IAmazonS3 _amazonS3;

    /// <summary>
    /// Constructor for the S3ActionsWrapper.
    /// </summary>
    /// <param name="amazonS3">The injected S3 client.</param>
    public S3ActionsWrapper(IAmazonS3 amazonS3, IConfiguration configuration)
    {
        _amazonS3 = amazonS3;
    }

    /// <summary>
    /// Create a new Amazon S3 bucket with object lock actions.
    /// </summary>
    /// <param name="bucketName">The name of the bucket to create.</param>
    /// <param name="enableObjectLock">True to enable object lock on the
 bucket.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> CreateBucketWithObjectLock(string bucketName, bool
 enableObjectLock)
    {
        Console.WriteLine($"\tCreating bucket {bucketName} with object lock
 {enableObjectLock}.");
        try
        {
            var request = new PutBucketRequest
            {
                BucketName = bucketName,
                UseClientRegion = true,
```

```
                ObjectLockEnabledForBucket = enableObjectLock,
            };

            var response = await _amazonS3.PutBucketAsync(request);

            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"Error creating bucket: '{ex.Message}'");
            return false;
        }
    }

    /// <summary>
    /// Enable object lock on an existing bucket.
    /// </summary>
    /// <param name="bucketName">The name of the bucket to modify.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> EnableObjectLockOnBucket(string bucketName)
    {
        try
        {
            // First, enable Versioning on the bucket.
            await _amazonS3.PutBucketVersioningAsync(new
PutBucketVersioningRequest()
            {
                BucketName = bucketName,
                VersioningConfig = new S3BucketVersioningConfig()
                {
                    EnableMfaDelete = false,
                    Status = VersionStatus.Enabled
                }
            });

            var request = new PutObjectLockConfigurationRequest()
            {
                BucketName = bucketName,
                ObjectLockConfiguration = new ObjectLockConfiguration()
                {
                    ObjectLockEnabled = new ObjectLockEnabled("Enabled"),
                },
            };
```

```
            var response = await
_amazonS3.PutObjectLockConfigurationAsync(request);
            Console.WriteLine($"\tAdded an object lock policy to bucket
{bucketName}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"Error modifying object lock: '{ex.Message}'");
            return false;
        }
    }

    /// <summary>
    /// Set or modify a retention period on an object in an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The key of the object.</param>
    /// <param name="retention">The retention mode.</param>
    /// <param name="retainUntilDate">The date retention expires.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ModifyObjectRetentionPeriod(string bucketName,
        string objectKey, ObjectLockRetentionMode retention, DateTime
retainUntilDate)
    {
        try
        {
            var request = new PutObjectRetentionRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
                Retention = new ObjectLockRetention()
                {
                    Mode = retention,
                    RetainUntilDate = retainUntilDate
                }
            };

            var response = await _amazonS3.PutObjectRetentionAsync(request);
            Console.WriteLine($"\tSet retention for {objectKey} in {bucketName}
until {retainUntilDate:d}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
```

```
        {
            Console.WriteLine($"\tError modifying retention period:
'{ex.Message}'");
            return false;
        }
    }

    /// <summary>
    /// Set or modify a retention period on an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket to modify.</param>
    /// <param name="retention">The retention mode.</param>
    /// <param name="retainUntilDate">The date for retention until.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ModifyBucketDefaultRetention(string bucketName, bool
enableObjectLock, ObjectLockRetentionMode retention, DateTime retainUntilDate)
    {
        var enabledString = enableObjectLock ? "Enabled" : "Disabled";
        var timeDifference = retainUntilDate.Subtract(DateTime.Now);
        try
        {
            // First, enable Versioning on the bucket.
            await _amazonS3.PutBucketVersioningAsync(new
PutBucketVersioningRequest()
            {
                BucketName = bucketName,
                VersioningConfig = new S3BucketVersioningConfig()
                {
                    EnableMfaDelete = false,
                    Status = VersionStatus.Enabled
                }
            });

            var request = new PutObjectLockConfigurationRequest()
            {
                BucketName = bucketName,
                ObjectLockConfiguration = new ObjectLockConfiguration()
                {
                    ObjectLockEnabled = new ObjectLockEnabled(enabledString),
                    Rule = new ObjectLockRule()
                    {
                        DefaultRetention = new DefaultRetention()
                        {
                            Mode = retention,
```

```
                            Days = timeDifference.Days // Can be specified in
days or years but not both.
                        }
                    }
                }
            };

            var response = await
_amazonS3.PutObjectLockConfigurationAsync(request);
            Console.WriteLine($"\tAdded a default retention to bucket
{bucketName}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tError modifying object lock: '{ex.Message}'");
            return false;
        }
    }

    /// <summary>
    /// Get the retention period for an S3 object.
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The object key.</param>
    /// <returns>The object retention details.</returns>
    public async Task<ObjectLockRetention> GetObjectRetention(string bucketName,
        string objectKey)
    {
        try
        {
            var request = new GetObjectRetentionRequest()
            {
                BucketName = bucketName,
                Key = objectKey
            };

            var response = await _amazonS3.GetObjectRetentionAsync(request);
            Console.WriteLine($"\tObject retention for {objectKey} in
{bucketName}: " +
                                $"\n\t{response.Retention.Mode} until
{response.Retention.RetainUntilDate:d}.");
            return response.Retention;
        }
```

```
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to fetch object lock retention:
'{ex.Message}'");
            return new ObjectLockRetention();
        }
    }

    /// <summary>
    /// Set or modify a legal hold on an object in an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The key of the object.</param>
    /// <param name="holdStatus">The On or Off status for the legal hold.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ModifyObjectLegalHold(string bucketName,
        string objectKey, ObjectLockLegalHoldStatus holdStatus)
    {
        try
        {
            var request = new PutObjectLegalHoldRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
                LegalHold = new ObjectLockLegalHold()
                {
                    Status = holdStatus
                }
            };

            var response = await _amazonS3.PutObjectLegalHoldAsync(request);
            Console.WriteLine($"\tModified legal hold for {objectKey} in
{bucketName}.");
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tError modifying legal hold: '{ex.Message}'");
            return false;
        }
    }

    /// <summary>
    /// Get the legal hold details for an S3 object.
```

```
    /// </summary>
    /// <param name="bucketName">The bucket of the object.</param>
    /// <param name="objectKey">The object key.</param>
    /// <returns>The object legal hold details.</returns>
    public async Task<ObjectLockLegalHold> GetObjectLegalHold(string bucketName,
        string objectKey)
    {
        try
        {
            var request = new GetObjectLegalHoldRequest()
            {
                BucketName = bucketName,
                Key = objectKey
            };

            var response = await _amazonS3.GetObjectLegalHoldAsync(request);
            Console.WriteLine($"\tObject legal hold for {objectKey} in
{bucketName}: " +
                              $"\n\tStatus: {response.LegalHold.Status}");
            return response.LegalHold;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to fetch legal hold: '{ex.Message}'");
            return new ObjectLockLegalHold();
        }
    }

    /// <summary>
    /// Get the object lock configuration details for an S3 bucket.
    /// </summary>
    /// <param name="bucketName">The bucket to get details.</param>
    /// <returns>The bucket's object lock configuration details.</returns>
    public async Task<ObjectLockConfiguration>
GetBucketObjectLockConfiguration(string bucketName)
    {
        try
        {
            var request = new GetObjectLockConfigurationRequest()
            {
                BucketName = bucketName
            };
```

```
            var response = await
_amazonS3.GetObjectLockConfigurationAsync(request);
            Console.WriteLine($"\tBucket object lock config for {bucketName} in
{bucketName}: " +
                                $"\n\tEnabled:
{response.ObjectLockConfiguration.ObjectLockEnabled}" +
                                $"\n\tRule:
{response.ObjectLockConfiguration.Rule?.DefaultRetention}");

            return response.ObjectLockConfiguration;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to fetch object lock config:
'{ex.Message}'");
            return new ObjectLockConfiguration();
        }
    }

    /// <summary>
    /// Upload a file from the local computer to an Amazon S3 bucket.
    /// </summary>
    /// <param name="bucketName">The Amazon S3 bucket to use.</param>
    /// <param name="objectName">The object to upload.</param>
    /// <param name="filePath">The path, including file name, of the object to
upload.</param>
    /// <returns>True if success.<returns>
    public async Task<bool> UploadFileAsync(string bucketName, string objectName,
string filePath)
    {
        var request = new PutObjectRequest
        {
            BucketName = bucketName,
            Key = objectName,
            FilePath = filePath,
            ChecksumAlgorithm = ChecksumAlgorithm.SHA256
        };

        var response = await _amazonS3.PutObjectAsync(request);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            Console.WriteLine($"\tSuccessfully uploaded {objectName} to
{bucketName}.");
            return true;
```

```
        }
        else
        {
            Console.WriteLine($"\tCould not upload {objectName} to
 {bucketName}.");
            return false;
        }
    }

    /// <summary>
    /// List bucket objects and versions.
    /// </summary>
    /// <param name="bucketName">The Amazon S3 bucket to use.</param>
    /// <returns>The list of objects and versions.</returns>
    public async Task<ListVersionsResponse> ListBucketObjectsAndVersions(string
 bucketName)
    {
        var request = new ListVersionsRequest()
        {
            BucketName = bucketName
        };

        var response = await _amazonS3.ListVersionsAsync(request);
        return response;
    }

    /// <summary>
    /// Delete an object from a specific bucket.
    /// </summary>
    /// <param name="bucketName">The Amazon S3 bucket to use.</param>
    /// <param name="objectKey">The key of the object to delete.</param>
    /// <param name="hasRetention">True if the object has retention settings.</
param>
    /// <param name="versionId">Optional versionId.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteObjectFromBucket(string bucketName, string
 objectKey, bool hasRetention, string? versionId = null)
    {
        try
        {
            var request = new DeleteObjectRequest()
            {
                BucketName = bucketName,
                Key = objectKey,
```

```
                VersionId = versionId,
            };
            if (hasRetention)
            {
                // Set the BypassGovernanceRetention header
                // if the file has retention settings.
                request.BypassGovernanceRetention = true;
            }
            await _amazonS3.DeleteObjectAsync(request);
            Console.WriteLine(
                $"Deleted {objectKey} in {bucketName}.");
            return true;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to delete object {objectKey} in bucket
{bucketName}: " + ex.Message);
            return false;
        }
    }

    /// <summary>
    /// Delete a specific bucket.
    /// </summary>
    /// <param name="bucketName">The Amazon S3 bucket to use.</param>
    /// <param name="objectKey">The key of the object to delete.</param>
    /// <param name="versionId">Optional versionId.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> DeleteBucketByName(string bucketName)
    {
        try
        {
            var request = new DeleteBucketRequest() { BucketName = bucketName, };
            var response = await _amazonS3.DeleteBucketAsync(request);
            Console.WriteLine($"\tDelete for {bucketName} complete.");
            return response.HttpStatusCode == HttpStatusCode.OK;
        }
        catch (AmazonS3Exception ex)
        {
            Console.WriteLine($"\tUnable to delete bucket {bucketName}: " +
ex.Message);
            return false;
        }
```

```
    }

}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for .NET .
  - GetObjectLegalHold
  - GetObjectLockConfiguration
  - GetObjectRetention
  - PutObjectLegalHold
  - PutObjectLockConfiguration
  - PutObjectRetention

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Jalankan skenario interaktif yang mendemonstrasikan fitur kunci objek Amazon S3.

```
// ObjectLockScenario contains the steps to run the S3 Object Lock workflow.
type ObjectLockScenario struct {
 questioner demotools.IQuestioner
 resources  Resources
 s3Actions  *actions.S3Actions
 sdkConfig  aws.Config
}

// NewObjectLockScenario constructs a new ObjectLockScenario instance.
func NewObjectLockScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner) ObjectLockScenario {
 scenario := ObjectLockScenario{
  questioner: questioner,
```

```
  resources:  Resources{},
  s3Actions:  &actions.S3Actions{S3Client: s3.NewFromConfig(sdkConfig)},
  sdkConfig:  sdkConfig,
 }
 scenario.s3Actions.S3Manager = manager.NewUploader(scenario.s3Actions.S3Client)
 scenario.resources.init(scenario.s3Actions, questioner)
 return scenario
}

type nameLocked struct {
 name   string
 locked bool
}

var createInfo = []nameLocked{
 {"standard-bucket", false},
 {"lock-bucket", true},
 {"retention-bucket", false},
}

// CreateBuckets creates the S3 buckets required for the workflow.
func (scenario *ObjectLockScenario) CreateBuckets(ctx context.Context) {
 log.Println("Let's create some S3 buckets to use for this workflow.")
 success := false
 for !success {
  prefix := scenario.questioner.Ask(
   "This example creates three buckets. Enter a prefix to name your buckets
 (remember bucket names must be globally unique):")

  for _, info := range createInfo {
   bucketName, err := scenario.s3Actions.CreateBucketWithLock(ctx,
 fmt.Sprintf("%s.%s", prefix, info.name), scenario.sdkConfig.Region, info.locked)
   if err != nil {
    switch err.(type) {
    case *types.BucketAlreadyExists, *types.BucketAlreadyOwnedByYou:
     log.Printf("Couldn't create bucket %s.\n", bucketName)
    default:
     panic(err)
    }
    break
   }
   scenario.resources.demoBuckets[info.name] = &DemoBucket{
    name:       bucketName,
    objectKeys: []string{},
```

```
    }
    log.Printf("Created bucket %s.\n", bucketName)
   }

   if len(scenario.resources.demoBuckets) < len(createInfo) {
    scenario.resources.deleteBuckets(ctx)
   } else {
    success = true
   }
  }

  log.Println("S3 buckets created.")
  log.Println(strings.Repeat("-", 88))
}

// EnableLockOnBucket enables object locking on an existing bucket.
func (scenario *ObjectLockScenario) EnableLockOnBucket(ctx context.Context) {
 log.Println("\nA bucket can be configured to use object locking.")
 scenario.questioner.Ask("Press Enter to continue.")

 var err error
 bucket := scenario.resources.demoBuckets["retention-bucket"]
 err = scenario.s3Actions.EnableObjectLockOnBucket(ctx, bucket.name)
 if err != nil {
  switch err.(type) {
  case *types.NoSuchBucket:
   log.Printf("Couldn't enable object locking on bucket %s.\n", bucket.name)
  default:
   panic(err)
  }
 } else {
  log.Printf("Object locking enabled on bucket %s.", bucket.name)
 }

 log.Println(strings.Repeat("-", 88))
}

// SetDefaultRetentionPolicy sets a default retention governance policy on a
 bucket.
func (scenario *ObjectLockScenario) SetDefaultRetentionPolicy(ctx
 context.Context) {
 log.Println("\nA bucket can be configured to use object locking with a default
 retention period.")
```

```
 bucket := scenario.resources.demoBuckets["retention-bucket"]
 retentionPeriod := scenario.questioner.AskInt("Enter the default retention
 period in days: ")
 err := scenario.s3Actions.ModifyDefaultBucketRetention(ctx,
 bucket.name, types.ObjectLockEnabledEnabled, int32(retentionPeriod),
 types.ObjectLockRetentionModeGovernance)
 if err != nil {
  switch err.(type) {
  case *types.NoSuchBucket:
   log.Printf("Couldn't configure a default retention period on bucket %s.\n",
 bucket.name)
  default:
   panic(err)
  }
 } else {
  log.Printf("Default retention policy set on bucket %s with %d day retention
 period.", bucket.name, retentionPeriod)
  bucket.retentionEnabled = true
 }

 log.Println(strings.Repeat("-", 88))
}

// UploadTestObjects uploads test objects to the S3 buckets.
func (scenario *ObjectLockScenario) UploadTestObjects(ctx context.Context) {
 log.Println("Uploading test objects to S3 buckets.")

 for _, info := range createInfo {
  bucket := scenario.resources.demoBuckets[info.name]
  for i := 0; i < 2; i++ {
   key, err := scenario.s3Actions.UploadObject(ctx, bucket.name,
 fmt.Sprintf("example-%d", i),
    fmt.Sprintf("Example object content #%d in bucket %s.", i, bucket.name))
   if err != nil {
    switch err.(type) {
    case *types.NoSuchBucket:
     log.Printf("Couldn't upload %s to bucket %s.\n", key, bucket.name)
    default:
     panic(err)
    }
   } else {
    log.Printf("Uploaded %s to bucket %s.\n", key, bucket.name)
    bucket.objectKeys = append(bucket.objectKeys, key)
   }
```

```
  }
 }

 scenario.questioner.Ask("Test objects uploaded. Press Enter to continue.")
 log.Println(strings.Repeat("-", 88))
}

// SetObjectLockConfigurations sets object lock configurations on the test
 objects.
func (scenario *ObjectLockScenario) SetObjectLockConfigurations(ctx
 context.Context) {
 log.Println("Now let's set object lock configurations on individual objects.")

 buckets := []*DemoBucket{scenario.resources.demoBuckets["lock-bucket"],
 scenario.resources.demoBuckets["retention-bucket"]}
 for _, bucket := range buckets {
  for index, objKey := range bucket.objectKeys {
   switch index {
   case 0:
    if scenario.questioner.AskBool(fmt.Sprintf("\nDo you want to add a legal hold
 to %s in %s (y/n)? ", objKey, bucket.name), "y") {
     err := scenario.s3Actions.PutObjectLegalHold(ctx, bucket.name, objKey, "",
 types.ObjectLockLegalHoldStatusOn)
     if err != nil {
      switch err.(type) {
      case *types.NoSuchKey:
       log.Printf("Couldn't set legal hold on %s.\n", objKey)
      default:
       panic(err)
      }
     } else {
      log.Printf("Legal hold set on %s.\n", objKey)
     }
    }
   case 1:
    q := fmt.Sprintf("\nDo you want to add a 1 day Governance retention period to
 %s in %s?\n"+
     "Reminder: Only a user with the s3:BypassGovernanceRetention permission is
 able to delete this object\n"+
     "or its bucket until the retention period has expired. (y/n) ", objKey,
 bucket.name)
    if scenario.questioner.AskBool(q, "y") {
     err := scenario.s3Actions.PutObjectRetention(ctx, bucket.name, objKey,
 types.ObjectLockRetentionModeGovernance, 1)
```

```
     if err != nil {
      switch err.(type) {
      case *types.NoSuchKey:
       log.Printf("Couldn't set retention period on %s in %s.\n", objKey,
 bucket.name)
      default:
       panic(err)
      }
     } else {
      log.Printf("Retention period set to 1 for %s.", objKey)
      bucket.retentionEnabled = true
     }
    }
   }
  }
 }
 log.Println(strings.Repeat("-", 88))
}

const (
 ListAll = iota
 DeleteObject
 DeleteRetentionObject
 OverwriteObject
 ViewRetention
 ViewLegalHold
 Finish
)

// InteractWithObjects allows the user to interact with the objects and test the
 object lock configurations.
func (scenario *ObjectLockScenario) InteractWithObjects(ctx context.Context) {
 log.Println("Now you can interact with the objects to explore the object lock
 configurations.")
 interactiveChoices := []string{
  "List all objects and buckets.",
  "Attempt to delete an object.",
  "Attempt to delete an object with retention period bypass.",
  "Attempt to overwrite a file.",
  "View the retention settings for an object.",
  "View the legal hold settings for an object.",
  "Finish the workflow."}

 choice := ListAll
```

```
for choice != Finish {
 objList := scenario.GetAllObjects(ctx)
 objChoices := scenario.makeObjectChoiceList(objList)
 choice = scenario.questioner.AskChoice("Choose an action from the menu:\n",
interactiveChoices)
 switch choice {
 case ListAll:
  log.Println("The current objects in the example buckets are:")
  for _, objChoice := range objChoices {
   log.Println("\t", objChoice)
  }
 case DeleteObject, DeleteRetentionObject:
  objChoice := scenario.questioner.AskChoice("Enter the number of the object to
delete:\n", objChoices)
  obj := objList[objChoice]
  deleted, err := scenario.s3Actions.DeleteObject(ctx, obj.bucket, obj.key,
obj.versionId, choice == DeleteRetentionObject)
  if err != nil {
   switch err.(type) {
   case *types.NoSuchKey:
    log.Println("Nothing to delete.")
   default:
    panic(err)
   }
  } else if deleted {
   log.Printf("Object %s deleted.\n", obj.key)
  }
 case OverwriteObject:
  objChoice := scenario.questioner.AskChoice("Enter the number of the object to
overwrite:\n", objChoices)
  obj := objList[objChoice]
  _, err := scenario.s3Actions.UploadObject(ctx, obj.bucket, obj.key,
fmt.Sprintf("New content in object %s.", obj.key))
  if err != nil {
   switch err.(type) {
   case *types.NoSuchBucket:
    log.Println("Couldn't upload to nonexistent bucket.")
   default:
    panic(err)
   }
  } else {
   log.Printf("Uploaded new content to object %s.\n", obj.key)
  }
 case ViewRetention:
```

```
    objChoice := scenario.questioner.AskChoice("Enter the number of the object to
 view:\n", objChoices)
    obj := objList[objChoice]
    retention, err := scenario.s3Actions.GetObjectRetention(ctx, obj.bucket,
 obj.key)
    if err != nil {
     switch err.(type) {
     case *types.NoSuchKey:
      log.Printf("Can't get retention configuration for %s.\n", obj.key)
     default:
      panic(err)
     }
    } else if retention != nil {
     log.Printf("Object %s has retention mode %s until %v.\n", obj.key,
 retention.Mode, retention.RetainUntilDate)
    } else {
     log.Printf("Object %s does not have object retention configured.\n", obj.key)
    }
   case ViewLegalHold:
    objChoice := scenario.questioner.AskChoice("Enter the number of the object to
 view:\n", objChoices)
    obj := objList[objChoice]
    legalHold, err := scenario.s3Actions.GetObjectLegalHold(ctx, obj.bucket,
 obj.key, obj.versionId)
    if err != nil {
     switch err.(type) {
     case *types.NoSuchKey:
      log.Printf("Can't get legal hold configuration for %s.\n", obj.key)
     default:
      panic(err)
     }
    } else if legalHold != nil {
     log.Printf("Object %s has legal hold %v.", obj.key, *legalHold)
    } else {
     log.Printf("Object %s does not have legal hold configured.", obj.key)
    }
   case Finish:
    log.Println("Let's clean up.")
   }
   log.Println(strings.Repeat("-", 88))
  }
 }

 type BucketKeyVersionId struct {
```

```
 bucket    string
 key       string
 versionId string
}

// GetAllObjects gets the object versions in the example S3 buckets and returns
 them in a flattened list.
func (scenario *ObjectLockScenario) GetAllObjects(ctx context.Context)
 []BucketKeyVersionId {
 var objectList []BucketKeyVersionId
 for _, info := range createInfo {
  bucket := scenario.resources.demoBuckets[info.name]
  versions, err := scenario.s3Actions.ListObjectVersions(ctx, bucket.name)
  if err != nil {
   switch err.(type) {
   case *types.NoSuchBucket:
    log.Printf("Couldn't get object versions for %s.\n", bucket.name)
   default:
    panic(err)
   }
  } else {
   for _, version := range versions {
    objectList = append(objectList,
     BucketKeyVersionId{bucket: bucket.name, key: *version.Key, versionId:
 *version.VersionId})
   }
  }
 }
 return objectList
}

// makeObjectChoiceList makes the object version list into a list of strings that
 are displayed
// as choices.
func (scenario *ObjectLockScenario) makeObjectChoiceList(bucketObjects
 []BucketKeyVersionId) []string {
 choices := make([]string, len(bucketObjects))
 for i := 0; i < len(bucketObjects); i++ {
  choices[i] = fmt.Sprintf("%s in %s with VersionId %s.",
   bucketObjects[i].key, bucketObjects[i].bucket, bucketObjects[i].versionId)
 }
 return choices
}
```

```
// Run runs the S3 Object Lock workflow scenario.
func (scenario *ObjectLockScenario) Run(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   _, isMock := scenario.questioner.(*demotools.MockQuestioner)
   if isMock || scenario.questioner.AskBool("Do you want to see the full error
 message (y/n)?", "y") {
    log.Println(r)
   }
   scenario.resources.Cleanup(ctx)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the Amazon S3 Object Lock Workflow Scenario.")
 log.Println(strings.Repeat("-", 88))

 scenario.CreateBuckets(ctx)
 scenario.EnableLockOnBucket(ctx)
 scenario.SetDefaultRetentionPolicy(ctx)
 scenario.UploadTestObjects(ctx)
 scenario.SetObjectLockConfigurations(ctx)
 scenario.InteractWithObjects(ctx)

 scenario.resources.Cleanup(ctx)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Tentukan struct yang membungkus tindakan S3 yang digunakan dalam contoh ini.

```
// S3Actions wraps S3 service actions.
type S3Actions struct {
 S3Client  *s3.Client
 S3Manager *manager.Uploader
}
```

```
// CreateBucketWithLock creates a new S3 bucket with optional object locking
 enabled
// and waits for the bucket to exist before returning.
func (actor S3Actions) CreateBucketWithLock(ctx context.Context, bucket string,
 region string, enableObjectLock bool) (string, error) {
 input := &s3.CreateBucketInput{
  Bucket: aws.String(bucket),
  CreateBucketConfiguration: &types.CreateBucketConfiguration{
   LocationConstraint: types.BucketLocationConstraint(region),
  },
 }

 if enableObjectLock {
  input.ObjectLockEnabledForBucket = aws.Bool(true)
 }

 _, err := actor.S3Client.CreateBucket(ctx, input)
 if err != nil {
  var owned *types.BucketAlreadyOwnedByYou
  var exists *types.BucketAlreadyExists
  if errors.As(err, &owned) {
   log.Printf("You already own bucket %s.\n", bucket)
   err = owned
  } else if errors.As(err, &exists) {
   log.Printf("Bucket %s already exists.\n", bucket)
   err = exists
  }
 } else {
  err = s3.NewBucketExistsWaiter(actor.S3Client).Wait(
   ctx, &s3.HeadBucketInput{Bucket: aws.String(bucket)}, time.Minute)
  if err != nil {
   log.Printf("Failed attempt to wait for bucket %s to exist.\n", bucket)
  }
 }

 return bucket, err
}



// GetObjectLegalHold retrieves the legal hold status for an S3 object.
```

```
func (actor S3Actions) GetObjectLegalHold(ctx context.Context, bucket string, key
 string, versionId string) (*types.ObjectLockLegalHoldStatus, error) {
 var status *types.ObjectLockLegalHoldStatus
 input := &s3.GetObjectLegalHoldInput{
  Bucket:    aws.String(bucket),
  Key:       aws.String(key),
  VersionId: aws.String(versionId),
 }

 output, err := actor.S3Client.GetObjectLegalHold(ctx, input)
 if err != nil {
  var noSuchKeyErr *types.NoSuchKey
  var apiErr *smithy.GenericAPIError
  if errors.As(err, &noSuchKeyErr) {
   log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
   err = noSuchKeyErr
  } else if errors.As(err, &apiErr) {
   switch apiErr.ErrorCode() {
   case "NoSuchObjectLockConfiguration":
    log.Printf("Object %s does not have an object lock configuration.\n", key)
    err = nil
   case "InvalidRequest":
    log.Printf("Bucket %s does not have an object lock configuration.\n", bucket)
    err = nil
   }
  }
 } else {
  status = &output.LegalHold.Status
 }

 return status, err
}



// GetObjectLockConfiguration retrieves the object lock configuration for an S3
 bucket.
func (actor S3Actions) GetObjectLockConfiguration(ctx context.Context, bucket
 string) (*types.ObjectLockConfiguration, error) {
 var lockConfig *types.ObjectLockConfiguration
 input := &s3.GetObjectLockConfigurationInput{
  Bucket: aws.String(bucket),
 }
```

```
 output, err := actor.S3Client.GetObjectLockConfiguration(ctx, input)
 if err != nil {
  var noBucket *types.NoSuchBucket
  var apiErr *smithy.GenericAPIError
  if errors.As(err, &noBucket) {
   log.Printf("Bucket %s does not exist.\n", bucket)
   err = noBucket
  } else if errors.As(err, &apiErr) && apiErr.ErrorCode() ==
 "ObjectLockConfigurationNotFoundError" {
   log.Printf("Bucket %s does not have an object lock configuration.\n", bucket)
   err = nil
  }
 } else {
  lockConfig = output.ObjectLockConfiguration
 }

 return lockConfig, err
}



// GetObjectRetention retrieves the object retention configuration for an S3
 object.
func (actor S3Actions) GetObjectRetention(ctx context.Context, bucket string, key
 string) (*types.ObjectLockRetention, error) {
 var retention *types.ObjectLockRetention
 input := &s3.GetObjectRetentionInput{
  Bucket: aws.String(bucket),
  Key:    aws.String(key),
 }

 output, err := actor.S3Client.GetObjectRetention(ctx, input)
 if err != nil {
  var noKey *types.NoSuchKey
  var apiErr *smithy.GenericAPIError
  if errors.As(err, &noKey) {
   log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
   err = noKey
  } else if errors.As(err, &apiErr) {
   switch apiErr.ErrorCode() {
   case "NoSuchObjectLockConfiguration":
    err = nil
   case "InvalidRequest":
    log.Printf("Bucket %s does not have locking enabled.", bucket)
```

```
    err = nil
   }
  }
 } else {
  retention = output.Retention
 }

 return retention, err
}



// PutObjectLegalHold sets the legal hold configuration for an S3 object.
func (actor S3Actions) PutObjectLegalHold(ctx context.Context, bucket string, key
 string, versionId string, legalHoldStatus types.ObjectLockLegalHoldStatus) error
 {
 input := &s3.PutObjectLegalHoldInput{
  Bucket: aws.String(bucket),
  Key:    aws.String(key),
  LegalHold: &types.ObjectLockLegalHold{
   Status: legalHoldStatus,
  },
 }
 if versionId != "" {
  input.VersionId = aws.String(versionId)
 }

 _, err := actor.S3Client.PutObjectLegalHold(ctx, input)
 if err != nil {
  var noKey *types.NoSuchKey
  if errors.As(err, &noKey) {
   log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
   err = noKey
  }
 }

 return err
}



// ModifyDefaultBucketRetention modifies the default retention period of an
 existing bucket.
func (actor S3Actions) ModifyDefaultBucketRetention(
```

```
 ctx context.Context, bucket string, lockMode types.ObjectLockEnabled,
 retentionPeriod int32, retentionMode types.ObjectLockRetentionMode) error {

 input := &s3.PutObjectLockConfigurationInput{
  Bucket: aws.String(bucket),
  ObjectLockConfiguration: &types.ObjectLockConfiguration{
   ObjectLockEnabled: lockMode,
   Rule: &types.ObjectLockRule{
    DefaultRetention: &types.DefaultRetention{
     Days: aws.Int32(retentionPeriod),
     Mode: retentionMode,
    },
   },
  },
 }
 _, err := actor.S3Client.PutObjectLockConfiguration(ctx, input)
 if err != nil {
  var noBucket *types.NoSuchBucket
  if errors.As(err, &noBucket) {
   log.Printf("Bucket %s does not exist.\n", bucket)
   err = noBucket
  }
 }

 return err
}



// EnableObjectLockOnBucket enables object locking on an existing bucket.
func (actor S3Actions) EnableObjectLockOnBucket(ctx context.Context, bucket
 string) error {
 // Versioning must be enabled on the bucket before object locking is enabled.
 verInput := &s3.PutBucketVersioningInput{
  Bucket: aws.String(bucket),
  VersioningConfiguration: &types.VersioningConfiguration{
   MFADelete: types.MFADeleteDisabled,
   Status:    types.BucketVersioningStatusEnabled,
  },
 }
 _, err := actor.S3Client.PutBucketVersioning(ctx, verInput)
 if err != nil {
  var noBucket *types.NoSuchBucket
  if errors.As(err, &noBucket) {
```

```
    log.Printf("Bucket %s does not exist.\n", bucket)
    err = noBucket
   }
   return err
  }

  input := &s3.PutObjectLockConfigurationInput{
   Bucket: aws.String(bucket),
   ObjectLockConfiguration: &types.ObjectLockConfiguration{
    ObjectLockEnabled: types.ObjectLockEnabledEnabled,
   },
  }
  _, err = actor.S3Client.PutObjectLockConfiguration(ctx, input)
  if err != nil {
   var noBucket *types.NoSuchBucket
   if errors.As(err, &noBucket) {
    log.Printf("Bucket %s does not exist.\n", bucket)
    err = noBucket
   }
  }

  return err
}



// PutObjectRetention sets the object retention configuration for an S3 object.
func (actor S3Actions) PutObjectRetention(ctx context.Context, bucket string, key
 string, retentionMode types.ObjectLockRetentionMode, retentionPeriodDays int32)
 error {
 input := &s3.PutObjectRetentionInput{
  Bucket: aws.String(bucket),
  Key:    aws.String(key),
  Retention: &types.ObjectLockRetention{
   Mode:            retentionMode,
   RetainUntilDate: aws.Time(time.Now().AddDate(0, 0, int(retentionPeriodDays))),
  },
  BypassGovernanceRetention: aws.Bool(true),
 }

 _, err := actor.S3Client.PutObjectRetention(ctx, input)
 if err != nil {
  var noKey *types.NoSuchKey
  if errors.As(err, &noKey) {
```

```
    log.Printf("Object %s does not exist in bucket %s.\n", key, bucket)
    err = noKey
   }
  }

  return err
}



// UploadObject uses the S3 upload manager to upload an object to a bucket.
func (actor S3Actions) UploadObject(ctx context.Context, bucket string, key
 string, contents string) (string, error) {
 var outKey string
 input := &s3.PutObjectInput{
  Bucket:            aws.String(bucket),
  Key:               aws.String(key),
  Body:              bytes.NewReader([]byte(contents)),
  ChecksumAlgorithm: types.ChecksumAlgorithmSha256,
 }
 output, err := actor.S3Manager.Upload(ctx, input)
 if err != nil {
  var noBucket *types.NoSuchBucket
  if errors.As(err, &noBucket) {
   log.Printf("Bucket %s does not exist.\n", bucket)
   err = noBucket
  }
 } else {
  err := s3.NewObjectExistsWaiter(actor.S3Client).Wait(ctx, &s3.HeadObjectInput{
   Bucket: aws.String(bucket),
   Key:    aws.String(key),
  }, time.Minute)
  if err != nil {
   log.Printf("Failed attempt to wait for object %s to exist in %s.\n", key,
 bucket)
  } else {
   outKey = *output.Key
  }
 }
 return outKey, err
}
```

```
// ListObjectVersions lists all versions of all objects in a bucket.
func (actor S3Actions) ListObjectVersions(ctx context.Context, bucket string)
 ([]types.ObjectVersion, error) {
 var err error
 var output *s3.ListObjectVersionsOutput
 var versions []types.ObjectVersion
 input := &s3.ListObjectVersionsInput{Bucket: aws.String(bucket)}
 versionPaginator := s3.NewListObjectVersionsPaginator(actor.S3Client, input)
 for versionPaginator.HasMorePages() {
  output, err = versionPaginator.NextPage(ctx)
  if err != nil {
   var noBucket *types.NoSuchBucket
   if errors.As(err, &noBucket) {
    log.Printf("Bucket %s does not exist.\n", bucket)
    err = noBucket
   }
   break
  } else {
   versions = append(versions, output.Versions...)
  }
 }
 return versions, err
}



// DeleteObject deletes an object from a bucket.
func (actor S3Actions) DeleteObject(ctx context.Context, bucket string, key
 string, versionId string, bypassGovernance bool) (bool, error) {
 deleted := false
 input := &s3.DeleteObjectInput{
  Bucket: aws.String(bucket),
  Key:    aws.String(key),
 }
 if versionId != "" {
  input.VersionId = aws.String(versionId)
 }
 if bypassGovernance {
  input.BypassGovernanceRetention = aws.Bool(true)
 }
 _, err := actor.S3Client.DeleteObject(ctx, input)
 if err != nil {
  var noKey *types.NoSuchKey
  var apiErr *smithy.GenericAPIError
```

```
  if errors.As(err, &noKey) {
   log.Printf("Object %s does not exist in %s.\n", key, bucket)
   err = noKey
  } else if errors.As(err, &apiErr) {
   switch apiErr.ErrorCode() {
   case "AccessDenied":
    log.Printf("Access denied: cannot delete object %s from %s.\n", key, bucket)
    err = nil
   case "InvalidArgument":
    if bypassGovernance {
     log.Printf("You cannot specify bypass governance on a bucket without lock
 enabled.")
     err = nil
    }
   }
  }
 } else {
  deleted = true
 }
 return deleted, err
}



// DeleteObjects deletes a list of objects from a bucket.
func (actor S3Actions) DeleteObjects(ctx context.Context, bucket string, objects
 []types.ObjectIdentifier, bypassGovernance bool) error {
 if len(objects) == 0 {
  return nil
 }

 input := s3.DeleteObjectsInput{
  Bucket: aws.String(bucket),
  Delete: &types.Delete{
   Objects: objects,
   Quiet:   aws.Bool(true),
  },
 }
 if bypassGovernance {
  input.BypassGovernanceRetention = aws.Bool(true)
 }
 delOut, err := actor.S3Client.DeleteObjects(ctx, &input)
 if err != nil || len(delOut.Errors) > 0 {
  log.Printf("Error deleting objects from bucket %s.\n", bucket)
```

```
  if err != nil {
   var noBucket *types.NoSuchBucket
   if errors.As(err, &noBucket) {
    log.Printf("Bucket %s does not exist.\n", bucket)
    err = noBucket
   }
  } else if len(delOut.Errors) > 0 {
   for _, outErr := range delOut.Errors {
    log.Printf("%s: %s\n", *outErr.Key, *outErr.Message)
   }
   err = fmt.Errorf("%s", *delOut.Errors[0].Message)
  }
 }
 return err
}
```

Pembersihan sumber daya

```
// DemoBucket contains metadata for buckets used in this example.
type DemoBucket struct {
 name             string
 legalHold        bool
 retentionEnabled bool
 objectKeys       []string
}

// Resources keeps track of AWS resources created during the ObjectLockScenario
 and handles
// cleanup when the scenario finishes.
type Resources struct {
 demoBuckets map[string]*DemoBucket

 s3Actions  *actions.S3Actions
 questioner demotools.IQuestioner
}

// init initializes objects in the Resources struct.
func (resources *Resources) init(s3Actions *actions.S3Actions, questioner
 demotools.IQuestioner) {
```

```
 resources.s3Actions = s3Actions
 resources.questioner = questioner
 resources.demoBuckets = map[string]*DemoBucket{}
}

// Cleanup deletes all AWS resources created during the ObjectLockScenario.
func (resources *Resources) Cleanup(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong during cleanup.\n%v\n", r)
   log.Println("Use the AWS Management Console to remove any remaining resources
 " +
    "that were created for this scenario.")
  }
 }()

 wantDelete := resources.questioner.AskBool("Do you want to remove all of the AWS
 resources that were created "+
  "during this demo (y/n)?", "y")
 if !wantDelete {
  log.Println("Be sure to remove resources when you're done with them to avoid
 unexpected charges!")
  return
 }

 log.Println("Removing objects from S3 buckets and deleting buckets...")
 resources.deleteBuckets(ctx)
 //resources.deleteRetentionObjects(resources.retentionBucket,
 resources.retentionObjects)

 log.Println("Cleanup complete.")
}

// deleteBuckets empties and then deletes all buckets created during the
 ObjectLockScenario.
func (resources *Resources) deleteBuckets(ctx context.Context) {
 for _, info := range createInfo {
  bucket := resources.demoBuckets[info.name]
  resources.deleteObjects(ctx, bucket)
  _, err := resources.s3Actions.S3Client.DeleteBucket(ctx, &s3.DeleteBucketInput{
   Bucket: aws.String(bucket.name),
  })
  if err != nil {
   panic(err)
```

```
  }
 }
 resources.demoBuckets = map[string]*DemoBucket{}
}

// deleteObjects deletes all objects in the specified bucket.
func (resources *Resources) deleteObjects(ctx context.Context, bucket
 *DemoBucket) {
 lockConfig, err := resources.s3Actions.GetObjectLockConfiguration(ctx,
 bucket.name)
 if err != nil {
  panic(err)
 }
 versions, err := resources.s3Actions.ListObjectVersions(ctx, bucket.name)
 if err != nil {
  switch err.(type) {
  case *types.NoSuchBucket:
   log.Printf("No objects to get from %s.\n", bucket.name)
  default:
   panic(err)
  }
 }
 delObjects := make([]types.ObjectIdentifier, len(versions))
 for i, version := range versions {
  if lockConfig != nil && lockConfig.ObjectLockEnabled ==
 types.ObjectLockEnabledEnabled {
   status, err := resources.s3Actions.GetObjectLegalHold(ctx, bucket.name,
 *version.Key, *version.VersionId)
   if err != nil {
    switch err.(type) {
    case *types.NoSuchKey:
     log.Printf("Couldn't determine legal hold status for %s in %s.\n",
 *version.Key, bucket.name)
    default:
     panic(err)
    }
   } else if status != nil && *status == types.ObjectLockLegalHoldStatusOn {
    err = resources.s3Actions.PutObjectLegalHold(ctx, bucket.name, *version.Key,
 *version.VersionId, types.ObjectLockLegalHoldStatusOff)
    if err != nil {
     switch err.(type) {
     case *types.NoSuchKey:
      log.Printf("Couldn't turn off legal hold for %s in %s.\n", *version.Key,
 bucket.name)
```

```
      default:
       panic(err)
      }
     }
    }
   }
   delObjects[i] = types.ObjectIdentifier{Key: version.Key, VersionId:
  version.VersionId}
  }
  err = resources.s3Actions.DeleteObjects(ctx, bucket.name, delObjects,
  bucket.retentionEnabled)
  if err != nil {
   switch err.(type) {
   case *types.NoSuchBucket:
    log.Println("Nothing to delete.")
   default:
    panic(err)
   }
  }
 }
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Go .
  - GetObjectLegalHold
  - GetObjectLockConfiguration
  - GetObjectRetention
  - PutObjectLegalHold
  - PutObjectLockConfiguration
  - PutObjectRetention

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Jalankan skenario interaktif yang mendemonstrasikan fitur kunci objek Amazon S3.

```
import software.amazon.awssdk.services.s3.model.ObjectLockLegalHold;
import software.amazon.awssdk.services.s3.model.ObjectLockRetention;
import java.io.BufferedWriter;
import java.io.IOException;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
import java.util.stream.Collectors;

/*
 Before running this Java V2 code example, set up your development
 environment, including your credentials.

 For more information, see the following documentation topic:
 https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/setup.html

 This Java example performs the following tasks:
    1. Create test Amazon Simple Storage Service (S3) buckets with different lock
 policies.
    2. Upload sample objects to each bucket.
    3. Set some Legal Hold and Retention Periods on objects and buckets.
    4. Investigate lock policies by viewing settings or attempting to delete or
 overwrite objects.
    5. Clean up objects and buckets.
 */
public class S3ObjectLockWorkflow {

    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    static String bucketName;
    static S3LockActions s3LockActions;
    private static final List<String> bucketNames = new ArrayList<>();
    private static final List<String> fileNames = new ArrayList<>();

    public static void main(String[] args) {
        // Get the current date and time to ensure bucket name is unique.
        LocalDateTime currentTime = LocalDateTime.now();

        // Format the date and time as a string.
```

```
        DateTimeFormatter formatter =
DateTimeFormatter.ofPattern("yyyyMMddHHmmss");
        String timeStamp = currentTime.format(formatter);

        s3LockActions = new S3LockActions();
        bucketName = "bucket"+timeStamp;
        Scanner scanner = new Scanner(System.in);

        System.out.println(DASHES);
        System.out.println("Welcome to the Amazon Simple Storage Service (S3)
Object Locking Workflow Scenario.");
        System.out.println("Press Enter to continue...");
        scanner.nextLine();
        configurationSetup();
        System.out.println(DASHES);

        System.out.println(DASHES);
        setup();
        System.out.println("Setup is complete. Press Enter to continue...");
        scanner.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Lets present the user with choices.");
        System.out.println("Press Enter to continue...");
        scanner.nextLine();
        demoActionChoices() ;
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Would you like to clean up the resources? (y/n)");
        String delAns = scanner.nextLine().trim();
        if (delAns.equalsIgnoreCase("y")) {
            cleanup();
            System.out.println("Clean up is complete.");
        }

        System.out.println("Press Enter to continue...");
        scanner.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Amazon S3 Object Locking Workflow is complete.");
        System.out.println(DASHES);
```

```
    }

    // Present the user with the demo action choices.
    public static void demoActionChoices() {
        String[] choices = {
            "List all files in buckets.",
            "Attempt to delete a file.",
            "Attempt to delete a file with retention period bypass.",
            "Attempt to overwrite a file.",
            "View the object and bucket retention settings for a file.",
            "View the legal hold settings for a file.",
            "Finish the workflow."
        };

        int choice = 0;
        while (true) {
            System.out.println(DASHES);
            choice = getChoiceResponse("Explore the S3 locking features by
selecting one of the following choices:", choices);
            System.out.println(DASHES);
            System.out.println("You selected "+choices[choice]);
            switch (choice) {
                case 0 -> {
                    s3LockActions.listBucketsAndObjects(bucketNames, true);
                }

                case 1 -> {
                    System.out.println("Enter the number of the object to
delete:");
                    List<S3InfoObject> allFiles =
s3LockActions.listBucketsAndObjects(bucketNames, true);
                    List<String> fileKeys = allFiles.stream().map(f ->
f.getKeyName()).collect(Collectors.toList());
                    String[] fileKeysArray = fileKeys.toArray(new String[0]);
                    int fileChoice = getChoiceResponse(null, fileKeysArray);
                    String objectKey = fileKeys.get(fileChoice);
                    String bucketName = allFiles.get(fileChoice).getBucketName();
                    String version = allFiles.get(fileChoice).getVersion();
                    s3LockActions.deleteObjectFromBucket(bucketName, objectKey,
false, version);
                }

                case 2 -> {
```

```
                    System.out.println("Enter the number of the object to
delete:");
                    List<S3InfoObject> allFiles =
s3LockActions.listBucketsAndObjects(bucketNames, true);
                    List<String> fileKeys = allFiles.stream().map(f ->
f.getKeyName()).collect(Collectors.toList());
                    String[] fileKeysArray = fileKeys.toArray(new String[0]);
                    int fileChoice = getChoiceResponse(null, fileKeysArray);
                    String objectKey = fileKeys.get(fileChoice);
                    String bucketName = allFiles.get(fileChoice).getBucketName();
                    String version = allFiles.get(fileChoice).getVersion();
                    s3LockActions.deleteObjectFromBucket(bucketName, objectKey,
true, version);
                }

                case 3 -> {
                    System.out.println("Enter the number of the object to
overwrite:");
                    List<S3InfoObject> allFiles =
s3LockActions.listBucketsAndObjects(bucketNames, true);
                    List<String> fileKeys = allFiles.stream().map(f ->
f.getKeyName()).collect(Collectors.toList());
                    String[] fileKeysArray = fileKeys.toArray(new String[0]);
                    int fileChoice = getChoiceResponse(null, fileKeysArray);
                    String objectKey = fileKeys.get(fileChoice);
                    String bucketName = allFiles.get(fileChoice).getBucketName();

                    // Attempt to overwrite the file.
                    try (BufferedWriter writer = new BufferedWriter(new
java.io.FileWriter(objectKey))) {
                        writer.write("This is a modified text.");

                    } catch (IOException e) {
                        e.printStackTrace();
                    }
                    s3LockActions.uploadFile(bucketName, objectKey, objectKey);
                }

                case 4 -> {
                    System.out.println("Enter the number of the object to
overwrite:");
                    List<S3InfoObject> allFiles =
s3LockActions.listBucketsAndObjects(bucketNames, true);
```

```
                    List<String> fileKeys = allFiles.stream().map(f ->
f.getKeyName()).collect(Collectors.toList());
                    String[] fileKeysArray = fileKeys.toArray(new String[0]);
                    int fileChoice = getChoiceResponse(null, fileKeysArray);
                    String objectKey = fileKeys.get(fileChoice);
                    String bucketName = allFiles.get(fileChoice).getBucketName();
                    s3LockActions.getObjectRetention(bucketName, objectKey);
                }

                case 5 -> {
                    System.out.println("Enter the number of the object to
view:");
                    List<S3InfoObject> allFiles =
s3LockActions.listBucketsAndObjects(bucketNames, true);
                    List<String> fileKeys = allFiles.stream().map(f ->
f.getKeyName()).collect(Collectors.toList());
                    String[] fileKeysArray = fileKeys.toArray(new String[0]);
                    int fileChoice = getChoiceResponse(null, fileKeysArray);
                    String objectKey = fileKeys.get(fileChoice);
                    String bucketName = allFiles.get(fileChoice).getBucketName();
                    s3LockActions.getObjectLegalHold(bucketName, objectKey);
                    s3LockActions.getBucketObjectLockConfiguration(bucketName);
                }

                case 6 -> {
                    System.out.println("Exiting the workflow...");
                    return;
                }

                default -> {
                    System.out.println("Invalid choice. Please select again.");
                }
            }
        }
    }

    // Clean up the resources from the scenario.
    private static void cleanup() {
        List<S3InfoObject> allFiles =
s3LockActions.listBucketsAndObjects(bucketNames, false);
        for (S3InfoObject fileInfo : allFiles) {
            String bucketName = fileInfo.getBucketName();
            String key = fileInfo.getKeyName();
            String version = fileInfo.getVersion();
```

```
            if (bucketName.contains("lock-enabled") ||
(bucketName.contains("retention-after-creation"))) {
                ObjectLockLegalHold legalHold =
s3LockActions.getObjectLegalHold(bucketName, key);
                if (legalHold != null) {
                    String holdStatus = legalHold.status().name();
                    System.out.println(holdStatus);
                    if (holdStatus.compareTo("ON") == 0) {
                        s3LockActions.modifyObjectLegalHold(bucketName, key,
false);
                    }
                }
                // Check for a retention period.
                ObjectLockRetention retention =
s3LockActions.getObjectRetention(bucketName, key);
                boolean hasRetentionPeriod ;
                hasRetentionPeriod = retention != null;
                s3LockActions.deleteObjectFromBucket(bucketName,
key,hasRetentionPeriod, version);

            } else {
                System.out.println(bucketName +" objects do not have a legal
lock");
                s3LockActions.deleteObjectFromBucket(bucketName, key,false,
version);
            }
        }

        // Delete the buckets.
        System.out.println("Delete "+bucketName);
        for (String bucket : bucketNames){
            s3LockActions.deleteBucketByName(bucket);
        }
    }

    private static void setup() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("""
                For this workflow, we will use the AWS SDK for Java to create
several S3
                buckets and files to demonstrate working with S3 locking
features.
                """);
```

```
        System.out.println("S3 buckets can be created either with or without
object lock enabled.");
        System.out.println("Press Enter to continue...");
        scanner.nextLine();

        // Create three S3 buckets.
        s3LockActions.createBucketWithLockOptions(false, bucketNames.get(0));
        s3LockActions.createBucketWithLockOptions(true, bucketNames.get(1));
        s3LockActions.createBucketWithLockOptions(false, bucketNames.get(2));
        System.out.println("Press Enter to continue.");
        scanner.nextLine();

        System.out.println("Bucket "+bucketNames.get(2) +" will be configured to
use object locking with a default retention period.");
        s3LockActions.modifyBucketDefaultRetention(bucketNames.get(2));
        System.out.println("Press Enter to continue.");
        scanner.nextLine();

        System.out.println("Object lock policies can also be added to existing
buckets. For this example, we will use "+bucketNames.get(1));
        s3LockActions.enableObjectLockOnBucket(bucketNames.get(1));
        System.out.println("Press Enter to continue.");
        scanner.nextLine();

        // Upload some files to the buckets.
        System.out.println("Now let's add some test files:");
        String fileName = "exampleFile.txt";
        int fileCount = 2;
        try (BufferedWriter writer = new BufferedWriter(new
java.io.FileWriter(fileName))) {
            writer.write("This is a sample file for uploading to a bucket.");

        } catch (IOException e) {
            e.printStackTrace();
        }

        for (String bucketName : bucketNames){
            for (int i = 0; i < fileCount; i++) {
                // Get the file name without extension.
                String fileNameWithoutExtension =
java.nio.file.Paths.get(fileName).getFileName().toString();
                int extensionIndex = fileNameWithoutExtension.lastIndexOf('.');
                if (extensionIndex > 0) {
```

```
                    fileNameWithoutExtension =
fileNameWithoutExtension.substring(0, extensionIndex);
                }

                // Create the numbered file names.
                String numberedFileName = fileNameWithoutExtension + i +
getFileExtension(fileName);
                fileNames.add(numberedFileName);
                s3LockActions.uploadFile(bucketName, numberedFileName, fileName);
            }
        }

        String question = null;
        System.out.print("Press Enter to continue...");
        scanner.nextLine();
        System.out.println("Now we can set some object lock policies on
individual files:");
        for (String bucketName : bucketNames) {
            for (int i = 0; i < fileNames.size(); i++){

                // No modifications to the objects in the first bucket.
                if (!bucketName.equals(bucketNames.get(0))) {
                    String exampleFileName = fileNames.get(i);
                    switch (i) {
                        case 0 -> {
                            question = "Would you like to add a legal hold to " +
exampleFileName + " in " + bucketName + " (y/n)?";
                            System.out.println(question);
                            String ans = scanner.nextLine().trim();
                            if (ans.equalsIgnoreCase("y")) {
                                System.out.println("**** You have selected to put
a legal hold " + exampleFileName);

                                // Set a legal hold.
                                s3LockActions.modifyObjectLegalHold(bucketName,
exampleFileName, true);
                            }
                        }
                        case 1 -> {
                            """
                                Would you like to add a 1 day Governance
retention period to %s in %s (y/n)?
```

```
                                Reminder: Only a user with the
 s3:BypassGovernanceRetention permission will be able to delete this file or its
 bucket until the retention period has expired.
                                """.formatted(exampleFileName, bucketName);
                            System.out.println(question);
                            String ans2 = scanner.nextLine().trim();
                            if (ans2.equalsIgnoreCase("y")) {

 s3LockActions.modifyObjectRetentionPeriod(bucketName, exampleFileName);
                            }
                        }
                    }
                }
            }
        }
    }

    // Get file extension.
    private static String getFileExtension(String fileName) {
        int dotIndex = fileName.lastIndexOf('.');
        if (dotIndex > 0) {
            return fileName.substring(dotIndex);
        }
        return "";
    }

    public static void configurationSetup() {
        String noLockBucketName = bucketName + "-no-lock";
        String lockEnabledBucketName = bucketName + "-lock-enabled";
        String retentionAfterCreationBucketName = bucketName + "-retention-after-
creation";
        bucketNames.add(noLockBucketName);
        bucketNames.add(lockEnabledBucketName);
        bucketNames.add(retentionAfterCreationBucketName);
    }

    public static int getChoiceResponse(String question, String[] choices) {
        Scanner scanner = new Scanner(System.in);
        if (question != null) {
            System.out.println(question);
            for (int i = 0; i < choices.length; i++) {
                System.out.println("\t" + (i + 1) + ". " + choices[i]);
            }
        }
```

```
        int choiceNumber = 0;
        while (choiceNumber < 1 || choiceNumber > choices.length) {
            String choice = scanner.nextLine();
            try {
                choiceNumber = Integer.parseInt(choice);
            } catch (NumberFormatException e) {
                System.out.println("Invalid choice. Please enter a valid
 number.");
            }
        }

        return choiceNumber - 1;
    }
}
```

Kelas pembungkus untuk fungsi S3.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.BucketVersioningStatus;
import software.amazon.awssdk.services.s3.model.ChecksumAlgorithm;
import software.amazon.awssdk.services.s3.model.CreateBucketRequest;
import software.amazon.awssdk.services.s3.model.DefaultRetention;
import software.amazon.awssdk.services.s3.model.DeleteBucketRequest;
import software.amazon.awssdk.services.s3.model.DeleteObjectRequest;
import software.amazon.awssdk.services.s3.model.GetObjectLegalHoldRequest;
import software.amazon.awssdk.services.s3.model.GetObjectLegalHoldResponse;
import
 software.amazon.awssdk.services.s3.model.GetObjectLockConfigurationRequest;
import
 software.amazon.awssdk.services.s3.model.GetObjectLockConfigurationResponse;
import software.amazon.awssdk.services.s3.model.GetObjectRetentionRequest;
import software.amazon.awssdk.services.s3.model.GetObjectRetentionResponse;
import software.amazon.awssdk.services.s3.model.HeadBucketRequest;
import software.amazon.awssdk.services.s3.model.ListObjectVersionsRequest;
import software.amazon.awssdk.services.s3.model.ListObjectVersionsResponse;
import software.amazon.awssdk.services.s3.model.MFADelete;
import software.amazon.awssdk.services.s3.model.ObjectLockConfiguration;
import software.amazon.awssdk.services.s3.model.ObjectLockEnabled;
import software.amazon.awssdk.services.s3.model.ObjectLockLegalHold;
import software.amazon.awssdk.services.s3.model.ObjectLockLegalHoldStatus;
```

```
import software.amazon.awssdk.services.s3.model.ObjectLockRetention;
import software.amazon.awssdk.services.s3.model.ObjectLockRetentionMode;
import software.amazon.awssdk.services.s3.model.ObjectLockRule;
import software.amazon.awssdk.services.s3.model.PutBucketVersioningRequest;
import software.amazon.awssdk.services.s3.model.PutObjectLegalHoldRequest;
import
 software.amazon.awssdk.services.s3.model.PutObjectLockConfigurationRequest;
import software.amazon.awssdk.services.s3.model.PutObjectRequest;
import software.amazon.awssdk.services.s3.model.PutObjectResponse;
import software.amazon.awssdk.services.s3.model.PutObjectRetentionRequest;
import software.amazon.awssdk.services.s3.model.S3Exception;
import software.amazon.awssdk.services.s3.model.VersioningConfiguration;
import software.amazon.awssdk.services.s3.waiters.S3Waiter;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.time.Instant;
import java.time.ZoneId;
import java.time.ZonedDateTime;
import java.time.format.DateTimeFormatter;
import java.time.temporal.ChronoUnit;
import java.util.List;
import java.util.concurrent.atomic.AtomicInteger;
import java.util.stream.Collectors;

// Contains application logic for the Amazon S3 operations used in this workflow.
public class S3LockActions {

    private static S3Client getClient() {
        return S3Client.builder()
            .region(Region.US_EAST_1)
            .build();
    }

    // Set or modify a retention period on an object in an S3 bucket.
    public void modifyObjectRetentionPeriod(String bucketName, String objectKey)
 {
        // Calculate the instant one day from now.
        Instant futureInstant = Instant.now().plus(1, ChronoUnit.DAYS);

        // Convert the Instant to a ZonedDateTime object with a specific time
 zone.
        ZonedDateTime zonedDateTime =
 futureInstant.atZone(ZoneId.systemDefault());
```

```
        // Define a formatter for human-readable output.
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd
HH:mm:ss");

        // Format the ZonedDateTime object to a human-readable date string.
        String humanReadableDate = formatter.format(zonedDateTime);

        // Print the formatted date string.
        System.out.println("Formatted Date: " + humanReadableDate);
        ObjectLockRetention retention = ObjectLockRetention.builder()
            .mode(ObjectLockRetentionMode.GOVERNANCE)
            .retainUntilDate(futureInstant)
            .build();

        PutObjectRetentionRequest retentionRequest =
PutObjectRetentionRequest.builder()
            .bucket(bucketName)
            .key(objectKey)
            .retention(retention)
            .build();

        getClient().putObjectRetention(retentionRequest);
        System.out.println("Set retention for "+objectKey +" in " +bucketName +"
until "+ humanReadableDate +".");
    }

    // Get the legal hold details for an S3 object.
    public ObjectLockLegalHold getObjectLegalHold(String bucketName, String
objectKey) {
        try {
            GetObjectLegalHoldRequest legalHoldRequest =
GetObjectLegalHoldRequest.builder()
                .bucket(bucketName)
                .key(objectKey)
                .build();

            GetObjectLegalHoldResponse response =
getClient().getObjectLegalHold(legalHoldRequest);
            System.out.println("Object legal hold for " + objectKey + " in " +
bucketName +
                ":\n\tStatus: " + response.legalHold().status());
            return response.legalHold();

        } catch (S3Exception ex) {
```

```
            System.out.println("\tUnable to fetch legal hold: '" +
ex.getMessage() + "'");
        }

        return null;
    }

    // Create a new Amazon S3 bucket with object lock options.
    public void createBucketWithLockOptions(boolean enableObjectLock, String
bucketName) {
        S3Waiter s3Waiter = getClient().waiter();
        CreateBucketRequest bucketRequest = CreateBucketRequest.builder()
            .bucket(bucketName)
            .objectLockEnabledForBucket(enableObjectLock)
            .build();

        getClient().createBucket(bucketRequest);
        HeadBucketRequest bucketRequestWait = HeadBucketRequest.builder()
            .bucket(bucketName)
            .build();

        // Wait until the bucket is created and print out the response.
        s3Waiter.waitUntilBucketExists(bucketRequestWait);
        System.out.println(bucketName + " is ready");
    }

    public List<S3InfoObject> listBucketsAndObjects(List<String> bucketNames,
Boolean interactive) {
        AtomicInteger counter = new AtomicInteger(0); // Initialize counter.
        return bucketNames.stream()
            .flatMap(bucketName ->
listBucketObjectsAndVersions(bucketName).versions().stream()
                .map(version -> {
                    S3InfoObject s3InfoObject = new S3InfoObject();
                    s3InfoObject.setBucketName(bucketName);
                    s3InfoObject.setVersion(version.versionId());
                    s3InfoObject.setKeyName(version.key());
                    return s3InfoObject;
                }))
            .peek(s3InfoObject -> {
                int i = counter.incrementAndGet(); // Increment and get the
updated value.
                if (interactive) {
                    System.out.println(i + ": "+ s3InfoObject.getKeyName());
```

```
                    System.out.printf("%5s Bucket name: %s\n", "",
s3InfoObject.getBucketName());
                    System.out.printf("%5s Version: %s\n", "",
s3InfoObject.getVersion());
                }
            })
            .collect(Collectors.toList());
    }

    public ListObjectVersionsResponse listBucketObjectsAndVersions(String
bucketName) {
        ListObjectVersionsRequest versionsRequest =
ListObjectVersionsRequest.builder()
            .bucket(bucketName)
            .build();

        return getClient().listObjectVersions(versionsRequest);
    }

    // Set or modify a retention period on an S3 bucket.
    public void modifyBucketDefaultRetention(String bucketName) {
        VersioningConfiguration versioningConfiguration =
VersioningConfiguration.builder()
            .mfaDelete(MFADelete.DISABLED)
            .status(BucketVersioningStatus.ENABLED)
            .build();

        PutBucketVersioningRequest versioningRequest =
PutBucketVersioningRequest.builder()
            .bucket(bucketName)
            .versioningConfiguration(versioningConfiguration)
            .build();

        getClient().putBucketVersioning(versioningRequest);
        DefaultRetention rention = DefaultRetention.builder()
            .days(1)
            .mode(ObjectLockRetentionMode.GOVERNANCE)
            .build();

        ObjectLockRule lockRule = ObjectLockRule.builder()
            .defaultRetention(rention)
            .build();
```

```
        ObjectLockConfiguration objectLockConfiguration =
ObjectLockConfiguration.builder()
            .objectLockEnabled(ObjectLockEnabled.ENABLED)
            .rule(lockRule)
            .build();

        PutObjectLockConfigurationRequest putObjectLockConfigurationRequest =
PutObjectLockConfigurationRequest.builder()
            .bucket(bucketName)
            .objectLockConfiguration(objectLockConfiguration)
            .build();


getClient().putObjectLockConfiguration(putObjectLockConfigurationRequest) ;
        System.out.println("Added a default retention to bucket "+bucketName
+".");
    }

    // Enable object lock on an existing bucket.
    public void enableObjectLockOnBucket(String bucketName) {
        try {
            VersioningConfiguration versioningConfiguration =
VersioningConfiguration.builder()
                .status(BucketVersioningStatus.ENABLED)
                .build();

            PutBucketVersioningRequest putBucketVersioningRequest =
PutBucketVersioningRequest.builder()
                .bucket(bucketName)
                .versioningConfiguration(versioningConfiguration)
                .build();

            // Enable versioning on the bucket.
            getClient().putBucketVersioning(putBucketVersioningRequest);
            PutObjectLockConfigurationRequest request =
PutObjectLockConfigurationRequest.builder()
                .bucket(bucketName)
                .objectLockConfiguration(ObjectLockConfiguration.builder()
                    .objectLockEnabled(ObjectLockEnabled.ENABLED)
                    .build())
                .build();

            getClient().putObjectLockConfiguration(request);
```

```
            System.out.println("Successfully enabled object lock on
"+bucketName);

        } catch (S3Exception ex) {
            System.out.println("Error modifying object lock: '" + ex.getMessage()
+ "'");
        }
    }

    public void uploadFile(String bucketName, String objectName, String filePath)
{
        Path file = Paths.get(filePath);
        PutObjectRequest request = PutObjectRequest.builder()
            .bucket(bucketName)
            .key(objectName)
            .checksumAlgorithm(ChecksumAlgorithm.SHA256)
            .build();

        PutObjectResponse response = getClient().putObject(request, file);
        if (response != null) {
            System.out.println("\tSuccessfully uploaded " + objectName + " to " +
bucketName + ".");
        } else {
            System.out.println("\tCould not upload " + objectName + " to " +
bucketName + ".");
        }
    }

    // Set or modify a legal hold on an object in an S3 bucket.
    public void modifyObjectLegalHold(String bucketName, String objectKey,
boolean legalHoldOn) {
        ObjectLockLegalHold legalHold ;
        if (legalHoldOn) {
            legalHold = ObjectLockLegalHold.builder()
                .status(ObjectLockLegalHoldStatus.ON)
                .build();
        } else {
            legalHold = ObjectLockLegalHold.builder()
                .status(ObjectLockLegalHoldStatus.OFF)
                .build();
        }

        PutObjectLegalHoldRequest legalHoldRequest =
PutObjectLegalHoldRequest.builder()
```

```
            .bucket(bucketName)
            .key(objectKey)
            .legalHold(legalHold)
            .build();

        getClient().putObjectLegalHold(legalHoldRequest) ;
        System.out.println("Modified legal hold for "+ objectKey +" in
"+bucketName +".");
    }

    // Delete an object from a specific bucket.
    public void deleteObjectFromBucket(String bucketName, String objectKey,
boolean hasRetention, String versionId) {
        try {
            DeleteObjectRequest objectRequest;
            if (hasRetention) {
                objectRequest = DeleteObjectRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .versionId(versionId)
                    .bypassGovernanceRetention(true)
                    .build();
            } else {
                objectRequest = DeleteObjectRequest.builder()
                    .bucket(bucketName)
                    .key(objectKey)
                    .versionId(versionId)
                    .build();
            }

            getClient().deleteObject(objectRequest) ;
            System.out.println("The object was successfully deleted");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }

    // Get the retention period for an S3 object.
    public ObjectLockRetention getObjectRetention(String bucketName, String key){
        try {
            GetObjectRetentionRequest retentionRequest =
GetObjectRetentionRequest.builder()
                .bucket(bucketName)
```

```
                .key(key)
                .build();

            GetObjectRetentionResponse response =
getClient().getObjectRetention(retentionRequest);
            System.out.println("tObject retention for "+key +"
in "+ bucketName +": " + response.retention().mode() +" until "+
response.retention().retainUntilDate() +".");
            return response.retention();

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            return null;
        }
    }

    public void deleteBucketByName(String bucketName) {
        try {
            DeleteBucketRequest request = DeleteBucketRequest.builder()
                .bucket(bucketName)
                .build();

            getClient().deleteBucket(request);
            System.out.println(bucketName +" was deleted.");

        } catch (S3Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }

    // Get the object lock configuration details for an S3 bucket.
    public void getBucketObjectLockConfiguration(String bucketName) {
        GetObjectLockConfigurationRequest objectLockConfigurationRequest =
GetObjectLockConfigurationRequest.builder()
            .bucket(bucketName)
            .build();

        GetObjectLockConfigurationResponse response =
getClient().getObjectLockConfiguration(objectLockConfigurationRequest);
        System.out.println("Bucket object lock config for "+bucketName +":  ");
        System.out.println("\tEnabled:
"+response.objectLockConfiguration().objectLockEnabled());
        System.out.println("\tRule: "+
response.objectLockConfiguration().rule().defaultRetention());
```

```
    }
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Java 2.x .
  - GetObjectLegalHold
  - GetObjectLockConfiguration
  - GetObjectRetention
  - PutObjectLegalHold
  - PutObjectLockConfiguration
  - PutObjectRetention

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

index.js- Entrypoint untuk alur kerja. Ini mengatur semua langkah. Kunjungi GitHub untuk melihat detail implementasi untuk Skenario ScenarioInput, ScenarioOutput,, dan ScenarioAction.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import * as Scenarios from "@aws-doc-sdk-examples/lib/scenario/index.js";
import {
  exitOnFalse,
  loadState,
  saveState,
} from "@aws-doc-sdk-examples/lib/scenario/steps-common.js";

import { welcome, welcomeContinue } from "./welcome.steps.js";
import {
  confirmCreateBuckets,
  confirmPopulateBuckets,
```

```
  confirmSetLegalHoldFileEnabled,
  confirmSetLegalHoldFileRetention,
  confirmSetRetentionPeriodFileEnabled,
  confirmSetRetentionPeriodFileRetention,
  confirmUpdateLockPolicy,
  confirmUpdateRetention,
  createBuckets,
  createBucketsAction,
  populateBuckets,
  populateBucketsAction,
  setLegalHoldFileEnabledAction,
  setLegalHoldFileRetentionAction,
  setRetentionPeriodFileEnabledAction,
  setRetentionPeriodFileRetentionAction,
  updateLockPolicy,
  updateLockPolicyAction,
  updateRetention,
  updateRetentionAction,
} from "./setup.steps.js";

/**
 * @param {Scenarios} scenarios
 * @param {Record<string, any>} initialState
 */
export const getWorkflowStages = (scenarios, initialState = {}) => {
  const client = new S3Client({});

  return {
    deploy: new scenarios.Scenario(
      "S3 Object Locking - Deploy",
      [
        welcome(scenarios),
        welcomeContinue(scenarios),
        exitOnFalse(scenarios, "welcomeContinue"),
        createBuckets(scenarios),
        confirmCreateBuckets(scenarios),
        exitOnFalse(scenarios, "confirmCreateBuckets"),
        createBucketsAction(scenarios, client),
        updateRetention(scenarios),
        confirmUpdateRetention(scenarios),
        exitOnFalse(scenarios, "confirmUpdateRetention"),
        updateRetentionAction(scenarios, client),
        populateBuckets(scenarios),
        confirmPopulateBuckets(scenarios),
```

```
        exitOnFalse(scenarios, "confirmPopulateBuckets"),
        populateBucketsAction(scenarios, client),
        updateLockPolicy(scenarios),
        confirmUpdateLockPolicy(scenarios),
        exitOnFalse(scenarios, "confirmUpdateLockPolicy"),
        updateLockPolicyAction(scenarios, client),
        confirmSetLegalHoldFileEnabled(scenarios),
        setLegalHoldFileEnabledAction(scenarios, client),
        confirmSetRetentionPeriodFileEnabled(scenarios),
        setRetentionPeriodFileEnabledAction(scenarios, client),
        confirmSetLegalHoldFileRetention(scenarios),
        setLegalHoldFileRetentionAction(scenarios, client),
        confirmSetRetentionPeriodFileRetention(scenarios),
        setRetentionPeriodFileRetentionAction(scenarios, client),
        saveState,
      ],
      initialState,
    ),
    demo: new scenarios.Scenario(
      "S3 Object Locking - Demo",
      [loadState, replAction(scenarios, client)],
      initialState,
    ),
    clean: new scenarios.Scenario(
      "S3 Object Locking - Destroy",
      [
        loadState,
        confirmCleanup(scenarios),
        exitOnFalse(scenarios, "confirmCleanup"),
        cleanupAction(scenarios, client),
      ],
      initialState,
    ),
  };
};

// Call function if run directly
import { fileURLToPath } from "url";
import { S3Client } from "@aws-sdk/client-s3";
import { cleanupAction, confirmCleanup } from "./clean.steps.js";
import { replAction } from "./repl.steps.js";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  const objectLockingScenarios = getWorkflowStages(Scenarios);
```

```
  Scenarios.parseScenarioArgs(objectLockingScenarios);
}
```

welcome.steps.js- Output pesan selamat datang ke konsol.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
/**
 * @typedef {import("@aws-doc-sdk-examples/lib/scenario/index.js")} Scenarios
 */

/**
 * @param {Scenarios} scenarios
 */
const welcome = (scenarios) =>
  new scenarios.ScenarioOutput(
    "welcome",
    `Welcome to the Amazon Simple Storage Service (S3) Object Locking Workflow
 Scenario. For this workflow, we will use the AWS SDK for JavaScript to create
 several S3 buckets and files to demonstrate working with S3 locking features.`,
    { header: true },
  );

/**
 * @param {Scenarios} scenarios
 */
const welcomeContinue = (scenarios) =>
  new scenarios.ScenarioInput(
    "welcomeContinue",
    "Press Enter when you are ready to start.",
    { type: "confirm" },
  );

export { welcome, welcomeContinue };
```

setup.steps.js- Menyebarkan ember, objek, dan pengaturan file.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import {
  BucketVersioningStatus,
```

```
  ChecksumAlgorithm,
  CreateBucketCommand,
  MFADeleteStatus,
  PutBucketVersioningCommand,
  PutObjectCommand,
  PutObjectLockConfigurationCommand,
  PutObjectLegalHoldCommand,
  PutObjectRetentionCommand,
  ObjectLockLegalHoldStatus,
  ObjectLockRetentionMode,
} from "@aws-sdk/client-s3";

/**
 * @typedef {import("@aws-doc-sdk-examples/lib/scenario/index.js")} Scenarios
 */

/**
 * @typedef {import("@aws-sdk/client-s3").S3Client} S3Client
 */

const bucketPrefix = "js-object-locking";

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const createBuckets = (scenarios) =>
  new scenarios.ScenarioOutput(
    "createBuckets",
    `The following buckets will be created:
         ${bucketPrefix}-no-lock with object lock False.
         ${bucketPrefix}-lock-enabled with object lock True.
         ${bucketPrefix}-retention-after-creation with object lock False.`,
    { preformatted: true },
  );

/**
 * @param {Scenarios} scenarios
 */
const confirmCreateBuckets = (scenarios) =>
  new scenarios.ScenarioInput("confirmCreateBuckets", "Create the buckets?", {
    type: "confirm",
  });
```

```
/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const createBucketsAction = (scenarios, client) =>
  new scenarios.ScenarioAction("createBucketsAction", async (state) => {
    const noLockBucketName = `${bucketPrefix}-no-lock`;
    const lockEnabledBucketName = `${bucketPrefix}-lock-enabled`;
    const retentionBucketName = `${bucketPrefix}-retention-after-creation`;

    await client.send(new CreateBucketCommand({ Bucket: noLockBucketName }));
    await client.send(
      new CreateBucketCommand({
        Bucket: lockEnabledBucketName,
        ObjectLockEnabledForBucket: true,
      }),
    );
    await client.send(new CreateBucketCommand({ Bucket: retentionBucketName }));

    state.noLockBucketName = noLockBucketName;
    state.lockEnabledBucketName = lockEnabledBucketName;
    state.retentionBucketName = retentionBucketName;
  });

/**
 * @param {Scenarios} scenarios
 */
const populateBuckets = (scenarios) =>
  new scenarios.ScenarioOutput(
    "populateBuckets",
    `The following test files will be created:
         file0.txt in ${bucketPrefix}-no-lock.
         file1.txt in ${bucketPrefix}-no-lock.
         file0.txt in ${bucketPrefix}-lock-enabled.
         file1.txt in ${bucketPrefix}-lock-enabled.
         file0.txt in ${bucketPrefix}-retention-after-creation.
         file1.txt in ${bucketPrefix}-retention-after-creation.`,
    { preformatted: true },
  );

/**
 * @param {Scenarios} scenarios
 */
const confirmPopulateBuckets = (scenarios) =>
```

```
  new scenarios.ScenarioInput(
    "confirmPopulateBuckets",
    "Populate the buckets?",
    { type: "confirm" },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const populateBucketsAction = (scenarios, client) =>
  new scenarios.ScenarioAction("populateBucketsAction", async (state) => {
    await client.send(
      new PutObjectCommand({
        Bucket: state.noLockBucketName,
        Key: "file0.txt",
        Body: "Content",
        ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
      }),
    );
    await client.send(
      new PutObjectCommand({
        Bucket: state.noLockBucketName,
        Key: "file1.txt",
        Body: "Content",
        ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
      }),
    );
    await client.send(
      new PutObjectCommand({
        Bucket: state.lockEnabledBucketName,
        Key: "file0.txt",
        Body: "Content",
        ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
      }),
    );
    await client.send(
      new PutObjectCommand({
        Bucket: state.lockEnabledBucketName,
        Key: "file1.txt",
        Body: "Content",
        ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
      }),
    );
```

```
    await client.send(
      new PutObjectCommand({
        Bucket: state.retentionBucketName,
        Key: "file0.txt",
        Body: "Content",
        ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
      }),
    );
    await client.send(
      new PutObjectCommand({
        Bucket: state.retentionBucketName,
        Key: "file1.txt",
        Body: "Content",
        ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
      }),
    );
  });

/**
 * @param {Scenarios} scenarios
 */
const updateRetention = (scenarios) =>
  new scenarios.ScenarioOutput(
    "updateRetention",
    `A bucket can be configured to use object locking with a default retention
 period.
   A default retention period will be configured for ${bucketPrefix}-retention-
after-creation.`,
    { preformatted: true },
  );

/**
 * @param {Scenarios} scenarios
 */
const confirmUpdateRetention = (scenarios) =>
  new scenarios.ScenarioInput(
    "confirmUpdateRetention",
    "Configure default retention period?",
    { type: "confirm" },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
```

```
 */
const updateRetentionAction = (scenarios, client) =>
  new scenarios.ScenarioAction("updateRetentionAction", async (state) => {
    await client.send(
      new PutBucketVersioningCommand({
        Bucket: state.retentionBucketName,
        VersioningConfiguration: {
          MFADelete: MFADeleteStatus.Disabled,
          Status: BucketVersioningStatus.Enabled,
        },
      }),
    );

    await client.send(
      new PutObjectLockConfigurationCommand({
        Bucket: state.retentionBucketName,
        ObjectLockConfiguration: {
          ObjectLockEnabled: "Enabled",
          Rule: {
            DefaultRetention: {
              Mode: "GOVERNANCE",
              Years: 1,
            },
          },
        },
      }),
    );
  });

/**
 * @param {Scenarios} scenarios
 */
const updateLockPolicy = (scenarios) =>
  new scenarios.ScenarioOutput(
    "updateLockPolicy",
    `Object lock policies can also be added to existing buckets.
   An object lock policy will be added to ${bucketPrefix}-lock-enabled.`,
    { preformatted: true },
  );

/**
 * @param {Scenarios} scenarios
 */
const confirmUpdateLockPolicy = (scenarios) =>
```

```
  new scenarios.ScenarioInput(
    "confirmUpdateLockPolicy",
    "Add object lock policy?",
    { type: "confirm" },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const updateLockPolicyAction = (scenarios, client) =>
  new scenarios.ScenarioAction("updateLockPolicyAction", async (state) => {
    await client.send(
      new PutObjectLockConfigurationCommand({
        Bucket: state.lockEnabledBucketName,
        ObjectLockConfiguration: {
          ObjectLockEnabled: "Enabled",
        },
      }),
    );
  });

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const confirmSetLegalHoldFileEnabled = (scenarios) =>
  new scenarios.ScenarioInput(
    "confirmSetLegalHoldFileEnabled",
    (state) =>
      `Would you like to add a legal hold to file0.txt in
 ${state.lockEnabledBucketName}?`,
    {
      type: "confirm",
    },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const setLegalHoldFileEnabledAction = (scenarios, client) =>
  new scenarios.ScenarioAction(
    "setLegalHoldFileEnabledAction",
```

```
    async (state) => {
      await client.send(
        new PutObjectLegalHoldCommand({
          Bucket: state.lockEnabledBucketName,
          Key: "file0.txt",
          LegalHold: {
            Status: ObjectLockLegalHoldStatus.ON,
          },
        }),
      );
      console.log(
        `Modified legal hold for file0.txt in ${state.lockEnabledBucketName}.`,
      );
    },
    { skipWhen: (state) => !state.confirmSetLegalHoldFileEnabled },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const confirmSetRetentionPeriodFileEnabled = (scenarios) =>
  new scenarios.ScenarioInput(
    "confirmSetRetentionPeriodFileEnabled",
    (state) =>
      `Would you like to add a 1 day Governance retention period to file1.txt in
 ${state.lockEnabledBucketName}?
Reminder: Only a user with the s3:BypassGovernanceRetention permission will be
 able to delete this file or its bucket until the retention period has expired.`,
    {
      type: "confirm",
    },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const setRetentionPeriodFileEnabledAction = (scenarios, client) =>
  new scenarios.ScenarioAction(
    "setRetentionPeriodFileEnabledAction",
    async (state) => {
      const retentionDate = new Date();
      retentionDate.setDate(retentionDate.getDate() + 1);
```

```
      await client.send(
        new PutObjectRetentionCommand({
          Bucket: state.lockEnabledBucketName,
          Key: "file1.txt",
          Retention: {
            Mode: ObjectLockRetentionMode.GOVERNANCE,
            RetainUntilDate: retentionDate,
          },
        }),
      );
      console.log(
        `Set retention for file1.txt in ${state.lockEnabledBucketName} until
 ${retentionDate.toISOString().split("T")[0]}.`,
      );
    },
    { skipWhen: (state) => !state.confirmSetRetentionPeriodFileEnabled },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const confirmSetLegalHoldFileRetention = (scenarios) =>
  new scenarios.ScenarioInput(
    "confirmSetLegalHoldFileRetention",
    (state) =>
      `Would you like to add a legal hold to file0.txt in
 ${state.retentionBucketName}?`,
    {
      type: "confirm",
    },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const setLegalHoldFileRetentionAction = (scenarios, client) =>
  new scenarios.ScenarioAction(
    "setLegalHoldFileRetentionAction",
    async (state) => {
      await client.send(
        new PutObjectLegalHoldCommand({
          Bucket: state.retentionBucketName,
```

```
          Key: "file0.txt",
          LegalHold: {
            Status: ObjectLockLegalHoldStatus.ON,
          },
        }),
      );
      console.log(
        `Modified legal hold for file0.txt in ${state.retentionBucketName}.`,
      );
    },
    { skipWhen: (state) => !state.confirmSetLegalHoldFileRetention },
  );

/**
 * @param {Scenarios} scenarios
 */
const confirmSetRetentionPeriodFileRetention = (scenarios) =>
  new scenarios.ScenarioInput(
    "confirmSetRetentionPeriodFileRetention",
    (state) =>
      `Would you like to add a 1 day Governance retention period to file1.txt in
 ${state.retentionBucketName}?
Reminder: Only a user with the s3:BypassGovernanceRetention permission will be
 able to delete this file or its bucket until the retention period has expired.`,
    {
      type: "confirm",
    },
  );

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const setRetentionPeriodFileRetentionAction = (scenarios, client) =>
  new scenarios.ScenarioAction(
    "setRetentionPeriodFileRetentionAction",
    async (state) => {
      const retentionDate = new Date();
      retentionDate.setDate(retentionDate.getDate() + 1);
      await client.send(
        new PutObjectRetentionCommand({
          Bucket: state.retentionBucketName,
          Key: "file1.txt",
          Retention: {
```

```
            Mode: ObjectLockRetentionMode.GOVERNANCE,
            RetainUntilDate: retentionDate,
          },
          BypassGovernanceRetention: true,
        }),
      );
      console.log(
        `Set retention for file1.txt in ${state.retentionBucketName} until
 ${retentionDate.toISOString().split("T")[0]}.`,
      );
    },
    { skipWhen: (state) => !state.confirmSetRetentionPeriodFileRetention },
  );

export {
  createBuckets,
  confirmCreateBuckets,
  createBucketsAction,
  populateBuckets,
  confirmPopulateBuckets,
  populateBucketsAction,
  updateRetention,
  confirmUpdateRetention,
  updateRetentionAction,
  updateLockPolicy,
  confirmUpdateLockPolicy,
  updateLockPolicyAction,
  confirmSetLegalHoldFileEnabled,
  setLegalHoldFileEnabledAction,
  confirmSetRetentionPeriodFileEnabled,
  setRetentionPeriodFileEnabledAction,
  confirmSetLegalHoldFileRetention,
  setLegalHoldFileRetentionAction,
  confirmSetRetentionPeriodFileRetention,
  setRetentionPeriodFileRetentionAction,
};
```

repl.steps.js- Lihat dan hapus file di ember.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import {
```

```
  ChecksumAlgorithm,
  DeleteObjectCommand,
  GetObjectLegalHoldCommand,
  GetObjectLockConfigurationCommand,
  GetObjectRetentionCommand,
  ListObjectVersionsCommand,
  PutObjectCommand,
} from "@aws-sdk/client-s3";

/**
 * @typedef {import("@aws-doc-sdk-examples/lib/scenario/index.js")} Scenarios
 */

/**
 * @typedef {import("@aws-sdk/client-s3").S3Client} S3Client
 */

const choices = {
  EXIT: 0,
  LIST_ALL_FILES: 1,
  DELETE_FILE: 2,
  DELETE_FILE_WITH_RETENTION: 3,
  OVERWRITE_FILE: 4,
  VIEW_RETENTION_SETTINGS: 5,
  VIEW_LEGAL_HOLD_SETTINGS: 6,
};

/**
 * @param {Scenarios} scenarios
 */
const replInput = (scenarios) =>
  new scenarios.ScenarioInput(
    "replChoice",
    `Explore the S3 locking features by selecting one of the following choices`,
    {
      type: "select",
      choices: [
        { name: "List all files in buckets", value: choices.LIST_ALL_FILES },
        { name: "Attempt to delete a file.", value: choices.DELETE_FILE },
        {
          name: "Attempt to delete a file with retention period bypass.",
          value: choices.DELETE_FILE_WITH_RETENTION,
        },
        { name: "Attempt to overwrite a file.", value: choices.OVERWRITE_FILE },
```

```
        {
          name: "View the object and bucket retention settings for a file.",
          value: choices.VIEW_RETENTION_SETTINGS,
        },
        {
          name: "View the legal hold settings for a file.",
          value: choices.VIEW_LEGAL_HOLD_SETTINGS,
        },
        { name: "Finish the workflow.", value: choices.EXIT },
      ],
    },
  );

/**
 * @param {S3Client} client
 * @param {string[]} buckets
 */
const getAllFiles = async (client, buckets) => {
  /** @type {{bucket: string, key: string, version: string}[]} */
  const files = [];
  for (const bucket of buckets) {
    const objectsResponse = await client.send(
      new ListObjectVersionsCommand({ Bucket: bucket }),
    );
    for (const version of objectsResponse.Versions || []) {
      const { Key, VersionId } = version;
      files.push({ bucket, key: Key, version: VersionId });
    }
  }

  return files;
};

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const replAction = (scenarios, client) =>
  new scenarios.ScenarioAction(
    "replAction",
    async (state) => {
      const files = await getAllFiles(client, [
        state.noLockBucketName,
        state.lockEnabledBucketName,
```

```
        state.retentionBucketName,
      ]);

      const fileInput = new scenarios.ScenarioInput(
        "selectedFile",
        "Select a file:",
        {
          type: "select",
          choices: files.map((file, index) => ({
            name: `${index + 1}: ${file.bucket}: ${file.key} (version: ${
              file.version
            })`,
            value: index,
          })),
        },
      );

      const { replChoice } = state;

      switch (replChoice) {
        case choices.LIST_ALL_FILES: {
          const files = await getAllFiles(client, [
            state.noLockBucketName,
            state.lockEnabledBucketName,
            state.retentionBucketName,
          ]);
          state.replOutput = files
            .map(
              (file) =>
                `${file.bucket}: ${file.key} (version: ${file.version})`,
            )
            .join("\n");
          break;
        }
        case choices.DELETE_FILE: {
          /** @type {number} */
          const fileToDelete = await fileInput.handle(state);
          const selectedFile = files[fileToDelete];
          try {
            await client.send(
              new DeleteObjectCommand({
                Bucket: selectedFile.bucket,
                Key: selectedFile.key,
                VersionId: selectedFile.version,
```

```
            }),
          );
          state.replOutput = `Deleted ${selectedFile.key} in
${selectedFile.bucket}.`;
        } catch (err) {
          state.replOutput = `Unable to delete object ${selectedFile.key} in
bucket ${selectedFile.bucket}: ${err.message}`;
        }
        break;
      }
      case choices.DELETE_FILE_WITH_RETENTION: {
        /** @type {number} */
        const fileToDelete = await fileInput.handle(state);
        const selectedFile = files[fileToDelete];
        try {
          await client.send(
            new DeleteObjectCommand({
              Bucket: selectedFile.bucket,
              Key: selectedFile.key,
              VersionId: selectedFile.version,
              BypassGovernanceRetention: true,
            }),
          );
          state.replOutput = `Deleted ${selectedFile.key} in
${selectedFile.bucket}.`;
        } catch (err) {
          state.replOutput = `Unable to delete object ${selectedFile.key} in
bucket ${selectedFile.bucket}: ${err.message}`;
        }
        break;
      }
      case choices.OVERWRITE_FILE: {
        /** @type {number} */
        const fileToOverwrite = await fileInput.handle(state);
        const selectedFile = files[fileToOverwrite];
        try {
          await client.send(
            new PutObjectCommand({
              Bucket: selectedFile.bucket,
              Key: selectedFile.key,
              Body: "New content",
              ChecksumAlgorithm: ChecksumAlgorithm.SHA256,
            }),
          );
```

```
            state.replOutput = `Overwrote ${selectedFile.key} in
 ${selectedFile.bucket}.`;
          } catch (err) {
            state.replOutput = `Unable to overwrite object ${selectedFile.key} in
 bucket ${selectedFile.bucket}: ${err.message}`;
          }
          break;
        }
        case choices.VIEW_RETENTION_SETTINGS: {
          /** @type {number} */
          const fileToView = await fileInput.handle(state);
          const selectedFile = files[fileToView];
          try {
            const retention = await client.send(
              new GetObjectRetentionCommand({
                Bucket: selectedFile.bucket,
                Key: selectedFile.key,
                VersionId: selectedFile.version,
              }),
            );
            const bucketConfig = await client.send(
              new GetObjectLockConfigurationCommand({
                Bucket: selectedFile.bucket,
              }),
            );
            state.replOutput = `Object retention for ${selectedFile.key}
 in ${selectedFile.bucket}: ${retention.Retention?.Mode} until
 ${retention.Retention?.RetainUntilDate?.toISOString()}.
Bucket object lock config for ${selectedFile.bucket} in ${selectedFile.bucket}:
Enabled: ${bucketConfig.ObjectLockConfiguration?.ObjectLockEnabled}
Rule:
 ${JSON.stringify(bucketConfig.ObjectLockConfiguration?.Rule?.DefaultRetention)}`;
          } catch (err) {
            state.replOutput = `Unable to fetch object lock retention:
 '${err.message}'`;
          }
          break;
        }
        case choices.VIEW_LEGAL_HOLD_SETTINGS: {
          /** @type {number} */
          const fileToView = await fileInput.handle(state);
          const selectedFile = files[fileToView];
          try {
            const legalHold = await client.send(
```

```
              new GetObjectLegalHoldCommand({
                Bucket: selectedFile.bucket,
                Key: selectedFile.key,
                VersionId: selectedFile.version,
              }),
            );
            state.replOutput = `Object legal hold for ${selectedFile.key} in
 ${selectedFile.bucket}: Status: ${legalHold.LegalHold?.Status}`;
          } catch (err) {
            state.replOutput = `Unable to fetch legal hold: '${err.message}'`;
          }
          break;
        }
        default:
          throw new Error(`Invalid replChoice: ${replChoice}`);
      }
    },
    {
      whileConfig: {
        whileFn: ({ replChoice }) => replChoice !== choices.EXIT,
        input: replInput(scenarios),
        output: new scenarios.ScenarioOutput(
          "REPL output",
          (state) => state.replOutput,
          { preformatted: true },
        ),
      },
    },
  );

export { replInput, replAction, choices };
```

clean.steps.js- Hancurkan semua sumber daya yang dibuat.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import {
  DeleteObjectCommand,
  DeleteBucketCommand,
  ListObjectVersionsCommand,
  GetObjectLegalHoldCommand,
  GetObjectRetentionCommand,
```

```
  PutObjectLegalHoldCommand,
} from "@aws-sdk/client-s3";

/**
 * @typedef {import("@aws-doc-sdk-examples/lib/scenario/index.js")} Scenarios
 */

/**
 * @typedef {import("@aws-sdk/client-s3").S3Client} S3Client
 */

/**
 * @param {Scenarios} scenarios
 */
const confirmCleanup = (scenarios) =>
  new scenarios.ScenarioInput("confirmCleanup", "Clean up resources?", {
    type: "confirm",
  });

/**
 * @param {Scenarios} scenarios
 * @param {S3Client} client
 */
const cleanupAction = (scenarios, client) =>
  new scenarios.ScenarioAction("cleanupAction", async (state) => {
    const { noLockBucketName, lockEnabledBucketName, retentionBucketName } =
      state;

    const buckets = [
      noLockBucketName,
      lockEnabledBucketName,
      retentionBucketName,
    ];

    for (const bucket of buckets) {
      /** @type {import("@aws-sdk/client-s3").ListObjectVersionsCommandOutput} */
      let objectsResponse;

      try {
        objectsResponse = await client.send(
          new ListObjectVersionsCommand({
            Bucket: bucket,
          }),
        );
```

```
      } catch (e) {
        if (e instanceof Error && e.name === "NoSuchBucket") {
          console.log("Object's bucket has already been deleted.");
          continue;
        } else {
          throw e;
        }
      }

      for (const version of objectsResponse.Versions || []) {
        const { Key, VersionId } = version;

        try {
          const legalHold = await client.send(
            new GetObjectLegalHoldCommand({
              Bucket: bucket,
              Key,
              VersionId,
            }),
          );

          if (legalHold.LegalHold?.Status === "ON") {
            await client.send(
              new PutObjectLegalHoldCommand({
                Bucket: bucket,
                Key,
                VersionId,
                LegalHold: {
                  Status: "OFF",
                },
              }),
            );
          }
        } catch (err) {
          console.log(
            `Unable to fetch legal hold for ${Key} in ${bucket}:
'${err.message}'`,
          );
        }

        try {
          const retention = await client.send(
            new GetObjectRetentionCommand({
              Bucket: bucket,
```

```
              Key,
              VersionId,
            }),
          );

          if (retention.Retention?.Mode === "GOVERNANCE") {
            await client.send(
              new DeleteObjectCommand({
                Bucket: bucket,
                Key,
                VersionId,
                BypassGovernanceRetention: true,
              }),
            );
          }
        } catch (err) {
          console.log(
            `Unable to fetch object lock retention for ${Key} in ${bucket}:
 '${err.message}'`,
          );
        }

        await client.send(
          new DeleteObjectCommand({
            Bucket: bucket,
            Key,
            VersionId,
          }),
        );
      }

      await client.send(new DeleteBucketCommand({ Bucket: bucket }));
      console.log(`Delete for ${bucket} complete.`);
    }
  });

export { confirmCleanup, cleanupAction };
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for JavaScript .
  - GetObjectLegalHold
  - GetObjectLockConfiguration

- GetObjectRetention
- PutObjectLegalHold
- PutObjectLockConfiguration
- PutObjectRetention

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mengelola daftar kontrol akses (ACL) untuk bucket Amazon S3 menggunakan SDK AWS

Contoh kode berikut ini menunjukkan cara mengelola daftar kontrol akses (ACL) untuk bucket Amazon S3.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
using System;
using System.Collections.Generic;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// This example shows how to manage Amazon Simple Storage Service
/// (Amazon S3) access control lists (ACLs) to control Amazon S3 bucket
/// access.
/// </summary>
public class ManageACLs
{
    public static async Task Main()
```

```
        {
            string bucketName = "doc-example-bucket1";
            string newBucketName = "doc-example-bucket2";
            string keyName = "sample-object.txt";
            string emailAddress = "someone@example.com";

            // If the AWS Region where your bucket is located is different from
            // the Region defined for the default user, pass the Amazon S3
bucket's
            // name to the client constructor. It should look like this:
            // RegionEndpoint bucketRegion = RegionEndpoint.USEast1;
            IAmazonS3 client = new AmazonS3Client();

            await TestBucketObjectACLsAsync(client, bucketName, newBucketName,
keyName, emailAddress);
        }

        /// <summary>
        /// Creates a new Amazon S3 bucket with a canned ACL, then retrieves the
ACL
        /// information and then adds a new ACL to one of the objects in the
        /// Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
call
        /// methods to create a bucket, get an ACL, and add a different ACL to
        /// one of the objects.</param>
        /// <param name="bucketName">A string representing the original Amazon S3
        /// bucket name.</param>
        /// <param name="newBucketName">A string representing the name of the
        /// new bucket that will be created.</param>
        /// <param name="keyName">A string representing the key name of an Amazon
S3
        /// object for which we will change the ACL.</param>
        /// <param name="emailAddress">A string representing the email address
        /// belonging to the person to whom access to the Amazon S3 bucket will
be
        /// granted.</param>
        public static async Task TestBucketObjectACLsAsync(
            IAmazonS3 client,
            string bucketName,
            string newBucketName,
            string keyName,
            string emailAddress)
```

```
        {
            try
            {
                // Create a new Amazon S3 bucket and specify canned ACL.
                var success = await CreateBucketWithCannedACLAsync(client,
 newBucketName);

                // Get the ACL on a bucket.
                await GetBucketACLAsync(client, bucketName);

                // Add (replace) the ACL on an object in a bucket.
                await AddACLToExistingObjectAsync(client, bucketName, keyName,
 emailAddress);
            }
            catch (AmazonS3Exception amazonS3Exception)
            {
                Console.WriteLine($"Exception: {amazonS3Exception.Message}");
            }
        }

        /// <summary>
        /// Creates a new Amazon S3 bucket with a canned ACL attached.
        /// </summary>
        /// <param name="client">The initialized client object used to call
        /// PutBucketAsync.</param>
        /// <param name="newBucketName">A string representing the name of the
        /// new Amazon S3 bucket.</param>
        /// <returns>Returns a boolean value indicating success or failure.</
returns>
        public static async Task<bool> CreateBucketWithCannedACLAsync(IAmazonS3
 client, string newBucketName)
        {
            var request = new PutBucketRequest()
            {
                BucketName = newBucketName,
                BucketRegion = S3Region.EUWest1,

                // Add a canned ACL.
                CannedACL = S3CannedACL.LogDeliveryWrite,
            };

            var response = await client.PutBucketAsync(request);
            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
```

```
        /// <summary>
        /// Retrieves the ACL associated with the Amazon S3 bucket name in the
        /// bucketName parameter.
        /// </summary>
        /// <param name="client">The initialized client object used to call
        /// PutBucketAsync.</param>
        /// <param name="bucketName">The Amazon S3 bucket for which we want to
 get the
        /// ACL list.</param>
        /// <returns>Returns an S3AccessControlList returned from the call to
        /// GetACLAsync.</returns>
        public static async Task<S3AccessControlList> GetBucketACLAsync(IAmazonS3
 client, string bucketName)
        {
            GetACLResponse response = await client.GetACLAsync(new GetACLRequest
            {
                BucketName = bucketName,
            });

            return response.AccessControlList;
        }



        /// <summary>
        /// Adds a new ACL to an existing object in the Amazon S3 bucket.
        /// </summary>
        /// <param name="client">The initialized client object used to call
        /// PutBucketAsync.</param>
        /// <param name="bucketName">A string representing the name of the Amazon
 S3
        /// bucket containing the object to which we want to apply a new ACL.</
param>
        /// <param name="keyName">A string representing the name of the object
        /// to which we want to apply the new ACL.</param>
        /// <param name="emailAddress">The email address of the person to whom
        /// we will be applying to whom access will be granted.</param>
        public static async Task AddACLToExistingObjectAsync(IAmazonS3 client,
 string bucketName, string keyName, string emailAddress)
        {
            // Retrieve the ACL for an object.
```

```
            GetACLResponse aclResponse = await client.GetACLAsync(new
 GetACLRequest
            {
                BucketName = bucketName,
                Key = keyName,
            });

            S3AccessControlList acl = aclResponse.AccessControlList;

            // Retrieve the owner.
            Owner owner = acl.Owner;

            // Clear existing grants.
            acl.Grants.Clear();

            // Add a grant to reset the owner's full permission
            // (the previous clear statement removed all permissions).
            var fullControlGrant = new S3Grant
            {
                Grantee = new S3Grantee { CanonicalUser = acl.Owner.Id },
            };
            acl.AddGrant(fullControlGrant.Grantee, S3Permission.FULL_CONTROL);

            // Specify email to identify grantee for granting permissions.
            var grantUsingEmail = new S3Grant
            {
                Grantee = new S3Grantee { EmailAddress = emailAddress },
                Permission = S3Permission.WRITE_ACP,
            };

            // Specify log delivery group as grantee.
            var grantLogDeliveryGroup = new S3Grant
            {
                Grantee = new S3Grantee { URI = "http://acs.amazonaws.com/groups/
s3/LogDelivery" },
                Permission = S3Permission.WRITE,
            };

            // Create a new ACL.
            var newAcl = new S3AccessControlList
            {
                Grants = new List<S3Grant> { grantUsingEmail,
 grantLogDeliveryGroup },
                Owner = owner,
```

```
            };

            // Set the new ACL. We're throwing away the response here.
            _ = await client.PutACLAsync(new PutACLRequest
            {
                BucketName = bucketName,
                Key = keyName,
                AccessControlList = newAcl,
            });
        }

    }
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for .NET .
  - GetBucketAcl
  - GetObjectAcl
  - PutBucketAcl
  - PutObjectAcl

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mengelola objek Amazon S3 berversi dalam batch dengan fungsi Lambda menggunakan SDK AWS

Contoh kode berikut ini menunjukkan cara mengelola objek S3 berversi dalam batch dengan fungsi Lambda.

Python

SDK untuk Python (Boto3)

Menunjukkan cara memanipulasi objek berversi Amazon Simple Storage Service (Amazon S3) dalam batch dengan membuat pekerjaan yang memanggil fungsi untuk melakukan pemrosesan. AWS Lambda Contoh ini membuat bucket yang diaktifkan versinya, mengunggah

bait dari puisi You Are Old, Father William oleh Lewis Carroll, dan menggunakan pekerjaan batch Amazon S3 untuk menggubah puisi dengan berbagai cara.

Pelajari cara:

- Membuat fungsi Lambda yang beroperasi pada objek berversi.
- Membuat manifes objek untuk diperbarui.
- Membuat pekerjaan batch yang menginvokasi fungsi Lambda untuk memperbarui objek.
- Menghapus fungsi Lambda.
- Mengosongkan dan menghapus bucket berversi.

Contoh ini paling baik dilihat di GitHub. Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Amazon S3

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mengurai URI Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara mengurai URI Amazon S3 untuk mengekstrak komponen penting seperti nama bucket dan kunci objek.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Mengurai Amazon S3 URI dengan menggunakan kelas S3Uri.

```
import org.slf4j.Logger;
```

```
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.S3Uri;
import software.amazon.awssdk.services.s3.S3Utilities;

import java.net.URI;
import java.util.List;
import java.util.Map;

    /**
     *
     * @param s3Client    - An S3Client through which you acquire an S3Uri
 instance.
     * @param s3ObjectUrl - A complex URL (String) that is used to demonstrate
 S3Uri
     *                    capabilities.
     */
    public static void parseS3UriExample(S3Client s3Client, String s3ObjectUrl) {
        logger.info(s3ObjectUrl);
        // Console output:
        // 'https://s3.us-west-1.amazonaws.com/myBucket/resources/doc.txt?
versionId=abc123&partNumber=77&partNumber=88'.

        // Create an S3Utilities object using the configuration of the s3Client.
        S3Utilities s3Utilities = s3Client.utilities();

        // From a String URL create a URI object to pass to the parseUri()
 method.
        URI uri = URI.create(s3ObjectUrl);
        S3Uri s3Uri = s3Utilities.parseUri(uri);

        // If the URI contains no value for the Region, bucket or key, the SDK
 returns
        // an empty Optional.
        // The SDK returns decoded URI values.

        Region region = s3Uri.region().orElse(null);
        log("region", region);
        // Console output: 'region: us-west-1'.

        String bucket = s3Uri.bucket().orElse(null);
        log("bucket", bucket);
        // Console output: 'bucket: myBucket'.
```

```
        String key = s3Uri.key().orElse(null);
        log("key", key);
        // Console output: 'key: resources/doc.txt'.

        Boolean isPathStyle = s3Uri.isPathStyle();
        log("isPathStyle", isPathStyle);
        // Console output: 'isPathStyle: true'.

        // If the URI contains no query parameters, the SDK returns an empty map.
        Map<String, List<String>> queryParams = s3Uri.rawQueryParameters();
        log("rawQueryParameters", queryParams);
        // Console output: 'rawQueryParameters: {versionId=[abc123],
 partNumber=[77,
        // 88]}'.

        // Retrieve the first or all values for a query parameter as shown in the
        // following code.
        String versionId =
 s3Uri.firstMatchingRawQueryParameter("versionId").orElse(null);
        log("firstMatchingRawQueryParameter-versionId", versionId);
        // Console output: 'firstMatchingRawQueryParameter-versionId: abc123'.

        String partNumber =
 s3Uri.firstMatchingRawQueryParameter("partNumber").orElse(null);
        log("firstMatchingRawQueryParameter-partNumber", partNumber);
        // Console output: 'firstMatchingRawQueryParameter-partNumber: 77'.

        List<String> partNumbers =
 s3Uri.firstMatchingRawQueryParameters("partNumber");
        log("firstMatchingRawQueryParameter", partNumbers);
        // Console output: 'firstMatchingRawQueryParameter: [77, 88]'.

        /*
         * Object keys and query parameters with reserved or unsafe characters,
 must be
         * URL-encoded.
         * For example replace whitespace " " with "%20".
         * Valid:
         * "https://s3.us-west-1.amazonaws.com/myBucket/object%20key?query=
%5Bbrackets%5D"
         * Invalid:
         * "https://s3.us-west-1.amazonaws.com/myBucket/object key?
query=[brackets]"
```

```
         *
         * Virtual-hosted-style URIs with bucket names that contain a dot, ".",
the dot
         * must not be URL-encoded.
         * Valid: "https://my.Bucket.s3.us-west-1.amazonaws.com/key"
         * Invalid: "https://my%2EBucket.s3.us-west-1.amazonaws.com/key"
         */
    }

    private static void log(String s3UriElement, Object element) {
        if (element == null) {
            logger.info("{}: {}", s3UriElement, "null");
        } else {
            logger.info("{}: {}", s3UriElement, element);
        }
    }
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Melakukan salinan multi-bagian objek Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara melakukan penyalinan multibagian dari sebuah objek Amazon S3.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    using System;
    using System.Collections.Generic;
    using System.Threading.Tasks;
    using Amazon.S3;
```

```
using Amazon.S3.Model;

/// <summary>
/// This example shows how to perform a multi-part copy from one Amazon
/// Simple Storage Service (Amazon S3) bucket to another.
/// </summary>
public class MPUapiCopyObj
{
    private const string SourceBucket = "doc-example-bucket1";
    private const string TargetBucket = "doc-example-bucket2";
    private const string SourceObjectKey = "example.mov";
    private const string TargetObjectKey = "copied_video_file.mov";

    /// <summary>
    /// This method starts the multi-part upload.
    /// </summary>
    public static async Task Main()
    {
        var s3Client = new AmazonS3Client();
        Console.WriteLine("Copying object...");
        await MPUCopyObjectAsync(s3Client);
    }

    /// <summary>
    /// This method uses the passed client object to perform a multipart
    /// copy operation.
    /// </summary>
    /// <param name="client">An Amazon S3 client object that will be used
    /// to perform the copy.</param>
    public static async Task MPUCopyObjectAsync(AmazonS3Client client)
    {
        // Create a list to store the copy part responses.
        var copyResponses = new List<CopyPartResponse>();

        // Setup information required to initiate the multipart upload.
        var initiateRequest = new InitiateMultipartUploadRequest
        {
            BucketName = TargetBucket,
            Key = TargetObjectKey,
        };

        // Initiate the upload.
        InitiateMultipartUploadResponse initResponse =
            await client.InitiateMultipartUploadAsync(initiateRequest);
```

```
            // Save the upload ID.
            string uploadId = initResponse.UploadId;

            try
            {
                // Get the size of the object.
                var metadataRequest = new GetObjectMetadataRequest
                {
                    BucketName = SourceBucket,
                    Key = SourceObjectKey,
                };

                GetObjectMetadataResponse metadataResponse =
                    await client.GetObjectMetadataAsync(metadataRequest);
                var objectSize = metadataResponse.ContentLength; // Length in
bytes.

                // Copy the parts.
                var partSize = 5 * (long)Math.Pow(2, 20); // Part size is 5 MB.

                long bytePosition = 0;
                for (int i = 1; bytePosition < objectSize; i++)
                {
                    var copyRequest = new CopyPartRequest
                    {
                        DestinationBucket = TargetBucket,
                        DestinationKey = TargetObjectKey,
                        SourceBucket = SourceBucket,
                        SourceKey = SourceObjectKey,
                        UploadId = uploadId,
                        FirstByte = bytePosition,
                        LastByte = bytePosition + partSize - 1 >= objectSize ?
objectSize - 1 : bytePosition + partSize - 1,
                        PartNumber = i,
                    };

                    copyResponses.Add(await client.CopyPartAsync(copyRequest));

                    bytePosition += partSize;
                }

                // Set up to complete the copy.
                var completeRequest = new CompleteMultipartUploadRequest
```

```
                {
                    BucketName = TargetBucket,
                    Key = TargetObjectKey,
                    UploadId = initResponse.UploadId,
                };
                completeRequest.AddPartETags(copyResponses);

                // Complete the copy.
                CompleteMultipartUploadResponse completeUploadResponse =
                    await client.CompleteMultipartUploadAsync(completeRequest);
            }
            catch (AmazonS3Exception e)
            {
                Console.WriteLine($"Error encountered on server.
Message:'{e.Message}' when writing an object");
            }
            catch (Exception e)
            {
                Console.WriteLine($"Unknown encountered on server.
Message:'{e.Message}' when writing an object");
            }
        }
    }
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for .NET .
  - CompleteMultipartUpload
  - CreateMultipartUpload
  - GetObjectMetadata
  - UploadPartCopy

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Melakukan pengunggahan multi-bagian objek Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara melakukan pengunggahan multibagian ke objek Amazon S3.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Contoh kode menggunakan impor berikut.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.services.s3.S3AsyncClient;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.CompletedMultipartUpload;
import software.amazon.awssdk.services.s3.model.CompletedPart;
import software.amazon.awssdk.services.s3.model.CreateMultipartUploadResponse;
import software.amazon.awssdk.services.s3.model.PutObjectResponse;
import software.amazon.awssdk.services.s3.model.UploadPartRequest;
import software.amazon.awssdk.services.s3.model.UploadPartResponse;
import software.amazon.awssdk.services.s3.waiters.S3Waiter;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.FileUpload;
import software.amazon.awssdk.transfer.s3.model.UploadFileRequest;

import java.io.IOException;
import java.io.RandomAccessFile;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.ByteBuffer;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.List;
import java.util.Objects;
import java.util.UUID;
import java.util.concurrent.CompletableFuture;
```

Gunakan Manajer Transfer S3 di atas klien S3 AWS berbasis CRT untuk melakukan unggahan multibagian secara transparan ketika ukuran konten melebihi ambang batas. Ukuran ambang default adalah 8 MB.

```
    public void multipartUploadWithTransferManager(String filePath) {
        S3TransferManager transferManager = S3TransferManager.create();
        UploadFileRequest uploadFileRequest = UploadFileRequest.builder()
                .putObjectRequest(b -> b
                        .bucket(bucketName)
                        .key(key))
                .source(Paths.get(filePath))
                .build();
        FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);
        fileUpload.completionFuture().join();
        transferManager.close();
    }
```

Gunakan S3Client API untuk melakukan upload multipart.

```
    public void multipartUploadWithS3Client(String filePath) {

        // Initiate the multipart upload.
        CreateMultipartUploadResponse createMultipartUploadResponse =
s3Client.createMultipartUpload(b -> b
                .bucket(bucketName)
                .key(key));
        String uploadId = createMultipartUploadResponse.uploadId();

        // Upload the parts of the file.
        int partNumber = 1;
        List<CompletedPart> completedParts = new ArrayList<>();
        ByteBuffer bb = ByteBuffer.allocate(1024 * 1024 * 5); // 5 MB byte buffer

        try (RandomAccessFile file = new RandomAccessFile(filePath, "r")) {
            long fileSize = file.length();
            long position = 0;
            while (position < fileSize) {
                file.seek(position);
                long read = file.getChannel().read(bb);

                bb.flip(); // Swap position and limit before reading from the
buffer.
```

```
                UploadPartRequest uploadPartRequest = UploadPartRequest.builder()
                        .bucket(bucketName)
                        .key(key)
                        .uploadId(uploadId)
                        .partNumber(partNumber)
                        .build();

                UploadPartResponse partResponse = s3Client.uploadPart(
                        uploadPartRequest,
                        RequestBody.fromByteBuffer(bb));

                CompletedPart part = CompletedPart.builder()
                        .partNumber(partNumber)
                        .eTag(partResponse.eTag())
                        .build();
                completedParts.add(part);

                bb.clear();
                position += read;
                partNumber++;
            }
        } catch (IOException e) {
            logger.error(e.getMessage());
        }

        // Complete the multipart upload.
        s3Client.completeMultipartUpload(b -> b
                .bucket(bucketName)
                .key(key)
                .uploadId(uploadId)

.multipartUpload(CompletedMultipartUpload.builder().parts(completedParts).build()));
    }
```

Gunakan S3 AsyncClient API dengan dukungan multipart diaktifkan untuk melakukan upload multipart.

```
    public void multipartUploadWithS3AsyncClient(String filePath) {
        // Enable multipart support.
        S3AsyncClient s3AsyncClient = S3AsyncClient.builder()
                .multipartEnabled(true)
```

```
                .build();

        CompletableFuture<PutObjectResponse> response = s3AsyncClient.putObject(b
-> b
                        .bucket(bucketName)
                        .key(key),
                Paths.get(filePath));

        response.join();
        logger.info("File uploaded in multiple 8 MiB parts using
S3AsyncClient.");
    }
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Java 2.x .
  - CompleteMultipartUpload
  - CreateMultipartUpload
  - UploadPart

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Lacak unggahan atau unduh objek Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara melacak unggahan atau unduhan objek Amazon S3.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Lacak kemajuan unggahan file.

```
   public void trackUploadFile(S3TransferManager transferManager, String
bucketName,
                            String key, URI filePathURI) {
       UploadFileRequest uploadFileRequest = UploadFileRequest.builder()
               .putObjectRequest(b -> b.bucket(bucketName).key(key))
               .addTransferListener(LoggingTransferListener.create())  // Add
listener.
               .source(Paths.get(filePathURI))
               .build();

       FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);

       fileUpload.completionFuture().join();
       /*
           The SDK provides a LoggingTransferListener implementation of the
TransferListener interface.
           You can also implement the interface to provide your own logic.

           Configure log4J2 with settings such as the following.
               <Configuration status="WARN">
                   <Appenders>
                       <Console name="AlignedConsoleAppender"
target="SYSTEM_OUT">
                           <PatternLayout pattern="%m%n"/>
                       </Console>
                   </Appenders>

                   <Loggers>
                       <logger
name="software.amazon.awssdk.transfer.s3.progress.LoggingTransferListener"
level="INFO" additivity="false">
                           <AppenderRef ref="AlignedConsoleAppender"/>
                       </logger>
                   </Loggers>
               </Configuration>

           Log4J2 logs the progress. The following is example output for a 21.3
MB file upload.
               Transfer initiated...
               |                    | 0.0%
               |====                | 21.1%
               |============        | 60.5%
               |====================| 100.0%
```

```
                Transfer complete!
        */
    }
```

Lacak kemajuan unduhan file.

```
   public void trackDownloadFile(S3TransferManager transferManager, String
bucketName,
                            String key, String downloadedFileWithPath) {
       DownloadFileRequest downloadFileRequest = DownloadFileRequest.builder()
               .getObjectRequest(b -> b.bucket(bucketName).key(key))
               .addTransferListener(LoggingTransferListener.create())  // Add
listener.
               .destination(Paths.get(downloadedFileWithPath))
               .build();

       FileDownload downloadFile =
transferManager.downloadFile(downloadFileRequest);

       CompletedFileDownload downloadResult =
downloadFile.completionFuture().join();
       /*
           The SDK provides a LoggingTransferListener implementation of the
TransferListener interface.
           You can also implement the interface to provide your own logic.

           Configure log4J2 with settings such as the following.
               <Configuration status="WARN">
                   <Appenders>
                       <Console name="AlignedConsoleAppender"
target="SYSTEM_OUT">
                           <PatternLayout pattern="%m%n"/>
                       </Console>
                   </Appenders>

                   <Loggers>
                       <logger
name="software.amazon.awssdk.transfer.s3.progress.LoggingTransferListener"
level="INFO" additivity="false">
                           <AppenderRef ref="AlignedConsoleAppender"/>
                       </logger>
                   </Loggers>
```

```
                </Configuration>

            Log4J2 logs the progress. The following is example output for a 21.3
 MB file download.
                Transfer initiated...
                |=======             | 39.4%
                |===============     | 78.8%
                |====================| 100.0%
                Transfer complete!
        */
    }
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Java 2.x .
  - GetObject
  - PutObject

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Contoh pendekatan untuk pengujian unit dan integrasi dengan AWS SDK

Contoh kode berikut menunjukkan cara contoh teknik praktik terbaik saat menulis unit dan pengujian integrasi menggunakan AWS SDK.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Cargo.toml untuk contoh pengujian.

```
[package]
name = "testing-examples"
```

```
version = "0.1.0"
authors = [
  "John Disanti <jdisanti@amazon.com>",
  "Doug Schwartz <dougsch@amazon.com>",
]
edition = "2021"

# snippet-start:[testing.rust.Cargo.toml]
[dependencies]
async-trait = "0.1.51"
aws-config = { version = "1.0.1", features = ["behavior-version-latest"] }
aws-credential-types = { version = "1.0.1", features = [ "hardcoded-
credentials", ] }
aws-sdk-s3 = { version = "1.4.0" }
aws-smithy-types = { version = "1.0.1" }
aws-smithy-runtime = { version = "1.0.1", features = ["test-util"] }
aws-smithy-runtime-api = { version = "1.0.1", features = ["test-util"] }
aws-types = { version = "1.0.1" }
clap = { version = "~4.4", features = ["derive"] }
http = "0.2.9"
mockall = "0.11.4"
serde_json = "1"
tokio = { version = "1.20.1", features = ["full"] }
tracing-subscriber = { version = "0.3.15", features = ["env-filter"] }
# snippet-end:[testing.rust.Cargo.toml]

[[bin]]
name = "main"
path = "src/main.rs"
```

Contoh pengujian unit menggunakan automock dan pembungkus layanan.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

// snippet-start:[testing.rust.wrapper]
// snippet-start:[testing.rust.wrapper-uses]
use aws_sdk_s3 as s3;
#[allow(unused_imports)]
use mockall::automock;

use s3::operation::list_objects_v2::{ListObjectsV2Error, ListObjectsV2Output};
```

```
// snippet-end:[testing.rust.wrapper-uses]

// snippet-start:[testing.rust.wrapper-which-impl]
#[cfg(test)]
pub use MockS3Impl as S3;
#[cfg(not(test))]
pub use S3Impl as S3;
// snippet-end:[testing.rust.wrapper-which-impl]

// snippet-start:[testing.rust.wrapper-impl]
#[allow(dead_code)]
pub struct S3Impl {
    inner: s3::Client,
}

#[cfg_attr(test, automock)]
impl S3Impl {
    #[allow(dead_code)]
    pub fn new(inner: s3::Client) -> Self {
        Self { inner }
    }

    #[allow(dead_code)]
    pub async fn list_objects(
        &self,
        bucket: &str,
        prefix: &str,
        continuation_token: Option<String>,
    ) -> Result<ListObjectsV2Output, s3::error::SdkError<ListObjectsV2Error>> {
        self.inner
            .list_objects_v2()
            .bucket(bucket)
            .prefix(prefix)
            .set_continuation_token(continuation_token)
            .send()
            .await
    }
}
// snippet-end:[testing.rust.wrapper-impl]

// snippet-start:[testing.rust.wrapper-func]
#[allow(dead_code)]
pub async fn determine_prefix_file_size(
    // Now we take a reference to our trait object instead of the S3 client
```

```
    // s3_list: ListObjectsService,
    s3_list: S3,
    bucket: &str,
    prefix: &str,
) -> Result<usize, s3::Error> {
    let mut next_token: Option<String> = None;
    let mut total_size_bytes = 0;
    loop {
        let result = s3_list
            .list_objects(bucket, prefix, next_token.take())
            .await?;

        // Add up the file sizes we got back
        for object in result.contents() {
            total_size_bytes += object.size().unwrap_or(0) as usize;
        }

        // Handle pagination, and break the loop if there are no more pages
        next_token = result.next_continuation_token.clone();
        if next_token.is_none() {
            break;
        }
    }
    Ok(total_size_bytes)
}
// snippet-end:[testing.rust.wrapper-func]
// snippet-end:[testing.rust.wrapper]

// snippet-start:[testing.rust.wrapper-test-mod]
#[cfg(test)]
mod test {
    // snippet-start:[testing.rust.wrapper-tests]
    use super::*;
    use mockall::predicate::eq;

    // snippet-start:[testing.rust.wrapper-test-single]
    #[tokio::test]
    async fn test_single_page() {
        let mut mock = MockS3Impl::default();
        mock.expect_list_objects()
            .with(eq("test-bucket"), eq("test-prefix"), eq(None))
            .return_once(|_, _, _| {
                Ok(ListObjectsV2Output::builder()
                    .set_contents(Some(vec![
```

```
                        // Mock content for ListObjectsV2 response
                        s3::types::Object::builder().size(5).build(),
                        s3::types::Object::builder().size(2).build(),
                    ]))
                    .build())
            });

        // Run the code we want to test with it
        let size = determine_prefix_file_size(mock, "test-bucket", "test-prefix")
            .await
            .unwrap();

        // Verify we got the correct total size back
        assert_eq!(7, size);
    }
    // snippet-end:[testing.rust.wrapper-test-single]

    // snippet-start:[testing.rust.wrapper-test-multiple]
    #[tokio::test]
    async fn test_multiple_pages() {
        // Create the Mock instance with two pages of objects now
        let mut mock = MockS3Impl::default();
        mock.expect_list_objects()
            .with(eq("test-bucket"), eq("test-prefix"), eq(None))
            .return_once(|_, _, _| {
                Ok(ListObjectsV2Output::builder()
                    .set_contents(Some(vec![
                        // Mock content for ListObjectsV2 response
                        s3::types::Object::builder().size(5).build(),
                        s3::types::Object::builder().size(2).build(),
                    ]))
                    .set_next_continuation_token(Some("next".to_string()))
                    .build())
            });
        mock.expect_list_objects()
            .with(
                eq("test-bucket"),
                eq("test-prefix"),
                eq(Some("next".to_string())),
            )
            .return_once(|_, _, _| {
                Ok(ListObjectsV2Output::builder()
                    .set_contents(Some(vec![
                        // Mock content for ListObjectsV2 response
```

```
                        s3::types::Object::builder().size(3).build(),
                        s3::types::Object::builder().size(9).build(),
                    ]))
                    .build())
            });

        // Run the code we want to test with it
        let size = determine_prefix_file_size(mock, "test-bucket", "test-prefix")
            .await
            .unwrap();

        assert_eq!(19, size);
    }
    // snippet-end:[testing.rust.wrapper-test-multiple]
    // snippet-end:[testing.rust.wrapper-tests]
}
// snippet-end:[testing.rust.wrapper-test-mod]
```

Contoh pengujian integrasi menggunakan StaticReplayClient.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

// snippet-start:[testing.rust.replay-uses]
use aws_sdk_s3 as s3;
// snippet-end:[testing.rust.replay-uses]

#[allow(dead_code)]
// snippet-start:[testing.rust.replay]
pub async fn determine_prefix_file_size(
    // Now we take a reference to our trait object instead of the S3 client
    // s3_list: ListObjectsService,
    s3: s3::Client,
    bucket: &str,
    prefix: &str,
) -> Result<usize, s3::Error> {
    let mut next_token: Option<String> = None;
    let mut total_size_bytes = 0;
    loop {
        let result = s3
            .list_objects_v2()
            .prefix(prefix)
```

```
            .bucket(bucket)
            .set_continuation_token(next_token.take())
            .send()
            .await?;

        // Add up the file sizes we got back
        for object in result.contents() {
            total_size_bytes += object.size().unwrap_or(0) as usize;
        }

        // Handle pagination, and break the loop if there are no more pages
        next_token = result.next_continuation_token.clone();
        if next_token.is_none() {
            break;
        }
    }
    Ok(total_size_bytes)
}
// snippet-end:[testing.rust.replay]

#[allow(dead_code)]
// snippet-start:[testing.rust.replay-tests]
// snippet-start:[testing.rust.replay-make-credentials]
fn make_s3_test_credentials() -> s3::config::Credentials {
    s3::config::Credentials::new(
        "ATESTCLIENT",
        "astestsecretkey",
        Some("atestsessiontoken".to_string()),
        None,
        "",
    )
}
// snippet-end:[testing.rust.replay-make-credentials]

// snippet-start:[testing.rust.replay-test-module]
#[cfg(test)]
mod test {
    // snippet-start:[testing.rust.replay-test-single]
    use super::*;
    use aws_config::BehaviorVersion;
    use aws_sdk_s3 as s3;
    use aws_smithy_runtime::client::http::test_util::{ReplayEvent,
 StaticReplayClient};
    use aws_smithy_types::body::SdkBody;
```

```
    #[tokio::test]
    async fn test_single_page() {
        let page_1 = ReplayEvent::new(
                http::Request::builder()
                    .method("GET")
                    .uri("https://test-bucket.s3.us-east-1.amazonaws.com/?list-
type=2&prefix=test-prefix")
                    .body(SdkBody::empty())
                    .unwrap(),
                http::Response::builder()
                    .status(200)
                    .body(SdkBody::from(include_str!("./testing/
response_1.xml")))
                    .unwrap(),
            );
        let replay_client = StaticReplayClient::new(vec![page_1]);
        let client: s3::Client = s3::Client::from_conf(
            s3::Config::builder()
                .behavior_version(BehaviorVersion::latest())
                .credentials_provider(make_s3_test_credentials())
                .region(s3::config::Region::new("us-east-1"))
                .http_client(replay_client.clone())
                .build(),
        );

        // Run the code we want to test with it
        let size = determine_prefix_file_size(client, "test-bucket", "test-
prefix")
            .await
            .unwrap();

        // Verify we got the correct total size back
        assert_eq!(7, size);
        replay_client.assert_requests_match(&[]);
    }
    // snippet-end:[testing.rust.replay-test-single]

    // snippet-start:[testing.rust.replay-test-multiple]
    #[tokio::test]
    async fn test_multiple_pages() {
        // snippet-start:[testing.rust.replay-create-replay]
        let page_1 = ReplayEvent::new(
                http::Request::builder()
```

```
                    .method("GET")
                    .uri("https://test-bucket.s3.us-east-1.amazonaws.com/?list-
type=2&prefix=test-prefix")
                    .body(SdkBody::empty())
                    .unwrap(),
                http::Response::builder()
                    .status(200)
                    .body(SdkBody::from(include_str!("./testing/
response_multi_1.xml")))
                    .unwrap(),
            );
        let page_2 = ReplayEvent::new(
                http::Request::builder()
                    .method("GET")
                    .uri("https://test-bucket.s3.us-east-1.amazonaws.com/?list-
type=2&prefix=test-prefix&continuation-token=next")
                    .body(SdkBody::empty())
                    .unwrap(),
                http::Response::builder()
                    .status(200)
                    .body(SdkBody::from(include_str!("./testing/
response_multi_2.xml")))
                    .unwrap(),
            );
        let replay_client = StaticReplayClient::new(vec![page_1, page_2]);
        // snippet-end:[testing.rust.replay-create-replay]
        // snippet-start:[testing.rust.replay-create-client]
        let client: s3::Client = s3::Client::from_conf(
            s3::Config::builder()
                .behavior_version(BehaviorVersion::latest())
                .credentials_provider(make_s3_test_credentials())
                .region(s3::config::Region::new("us-east-1"))
                .http_client(replay_client.clone())
                .build(),
        );
        // snippet-end:[testing.rust.replay-create-client]

        // Run the code we want to test with it
        // snippet-start:[testing.rust.replay-test-and-verify]
        let size = determine_prefix_file_size(client, "test-bucket", "test-
prefix")
            .await
            .unwrap();
```

```
        assert_eq!(19, size);

        replay_client.assert_requests_match(&[]);
        // snippet-end:[testing.rust.replay-test-and-verify]
    }
    // snippet-end:[testing.rust.replay-test-multiple]
}
// snippet-end:[testing.rust.replay-tests]
// snippet-end:[testing.rust.replay-test-module]
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mengunggah direktori lokal secara rekursif ke bucket Amazon Simple Storage Service (Amazon S3)

Contoh kode berikut ini menunjukkan cara mengunggah direktori lokal secara rekursif ke bucket Amazon Simple Storage Service (Amazon S3).

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Gunakan S3 TransferManager untuk mengunggah direktori lokal. Lihat file lengkap dan lakukan pengujian.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.services.s3.model.ObjectIdentifier;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedDirectoryUpload;
import software.amazon.awssdk.transfer.s3.model.DirectoryUpload;
```

```
import software.amazon.awssdk.transfer.s3.model.UploadDirectoryRequest;

import java.net.URI;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.file.Paths;
import java.util.UUID;

    public Integer uploadDirectory(S3TransferManager transferManager,
            URI sourceDirectory, String bucketName) {
        DirectoryUpload directoryUpload =
 transferManager.uploadDirectory(UploadDirectoryRequest.builder()
                .source(Paths.get(sourceDirectory))
                .bucket(bucketName)
                .build());

        CompletedDirectoryUpload completedDirectoryUpload =
 directoryUpload.completionFuture().join();
        completedDirectoryUpload.failedTransfers()
                .forEach(fail -> logger.warn("Object [{}] failed to transfer",
 fail.toString()));
        return completedDirectoryUpload.failedTransfers().size();
    }
```

- Untuk detail API, lihat UploadDirectorydi Referensi AWS SDK for Java 2.x API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Unggah atau unduh file besar ke dan dari Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara mengunggah atau mengunduh file besar ke dan dari Amazon S3.

Untuk informasi selengkapnya, lihat Pengunggahan objek menggunakan unggahan multibagian.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Fungsi panggilan yang mentransfer file ke dan dari bucket S3 menggunakan Amazon TransferUtility S3.

```
global using System.Text;
global using Amazon.S3;
global using Amazon.S3.Model;
global using Amazon.S3.Transfer;
global using TransferUtilityBasics;



// This Amazon S3 client uses the default user credentials
// defined for this computer.
using Microsoft.Extensions.Configuration;

IAmazonS3 client = new AmazonS3Client();
var transferUtil = new TransferUtility(client);
IConfiguration _configuration;

_configuration = new ConfigurationBuilder()
    .SetBasePath(Directory.GetCurrentDirectory())
    .AddJsonFile("settings.json") // Load test settings from JSON file.
    .AddJsonFile("settings.local.json",
        true) // Optionally load local settings.
    .Build();

// Edit the values in settings.json to use an S3 bucket and files that
// exist on your AWS account and on the local computer where you
// run this scenario.
var bucketName = _configuration["BucketName"];
```

```
var localPath =
 $"{Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData)}\
\TransferFolder";

DisplayInstructions();

PressEnter();

Console.WriteLine();

// Upload a single file to an S3 bucket.
DisplayTitle("Upload a single file");

var fileToUpload = _configuration["FileToUpload"];
Console.WriteLine($"Uploading {fileToUpload} to the S3 bucket, {bucketName}.");

var success = await TransferMethods.UploadSingleFileAsync(transferUtil,
 bucketName, fileToUpload, localPath);
if (success)
{
    Console.WriteLine($"Successfully uploaded the file, {fileToUpload} to
 {bucketName}.");
}

PressEnter();

// Upload a local directory to an S3 bucket.
DisplayTitle("Upload all files from a local directory");
Console.WriteLine("Upload all the files in a local folder to an S3 bucket.");
const string keyPrefix = "UploadFolder";
var uploadPath = $"{localPath}\\UploadFolder";

Console.WriteLine($"Uploading the files in {uploadPath} to {bucketName}");
DisplayTitle($"{uploadPath} files");
DisplayLocalFiles(uploadPath);
Console.WriteLine();

PressEnter();

success = await TransferMethods.UploadFullDirectoryAsync(transferUtil,
 bucketName, keyPrefix, uploadPath);
if (success)
{
```

```
    Console.WriteLine($"Successfully uploaded the files in {uploadPath} to
 {bucketName}.");
    Console.WriteLine($"{bucketName} currently contains the following files:");
    await DisplayBucketFiles(client, bucketName, keyPrefix);
    Console.WriteLine();
}

PressEnter();

// Download a single file from an S3 bucket.
DisplayTitle("Download a single file");
Console.WriteLine("Now we will download a single file from an S3 bucket.");

var keyName = _configuration["FileToDownload"];

Console.WriteLine($"Downloading {keyName} from {bucketName}.");

success = await TransferMethods.DownloadSingleFileAsync(transferUtil, bucketName,
 keyName, localPath);
if (success)
{
    Console.WriteLine("$Successfully downloaded the file, {keyName} from
 {bucketName}.");
}

PressEnter();

// Download the contents of a directory from an S3 bucket.
DisplayTitle("Download the contents of an S3 bucket");
var s3Path = _configuration["S3Path"];
var downloadPath = $"{localPath}\\{s3Path}";

Console.WriteLine($"Downloading the contents of {bucketName}\\{s3Path}");
Console.WriteLine($"{bucketName}\\{s3Path} contains the following files:");
await DisplayBucketFiles(client, bucketName, s3Path);
Console.WriteLine();

success = await TransferMethods.DownloadS3DirectoryAsync(transferUtil,
 bucketName, s3Path, downloadPath);
if (success)
{
    Console.WriteLine($"Downloaded the files in {bucketName} to
 {downloadPath}.");
    Console.WriteLine($"{downloadPath} now contains the following files:");
```

```
    DisplayLocalFiles(downloadPath);
}

Console.WriteLine("\nThe TransferUtility Basics application has completed.");
PressEnter();

// Displays the title for a section of the scenario.
static void DisplayTitle(string titleText)
{
    var sepBar = new string('-', Console.WindowWidth);

    Console.WriteLine(sepBar);
    Console.WriteLine(CenterText(titleText));
    Console.WriteLine(sepBar);
}

// Displays a description of the actions to be performed by the scenario.
static void DisplayInstructions()
{
    var sepBar = new string('-', Console.WindowWidth);

    DisplayTitle("Amazon S3 Transfer Utility Basics");
    Console.WriteLine("This program shows how to use the Amazon S3 Transfer
 Utility.");
    Console.WriteLine("It performs the following actions:");
    Console.WriteLine("\t1. Upload a single object to an S3 bucket.");
    Console.WriteLine("\t2. Upload an entire directory from the local computer to
 an\n\t  S3 bucket.");
    Console.WriteLine("\t3. Download a single object from an S3 bucket.");
    Console.WriteLine("\t4. Download the objects in an S3 bucket to a local
 directory.");
    Console.WriteLine($"\n{sepBar}");
}

// Pauses the scenario.
static void PressEnter()
{
    Console.WriteLine("Press <Enter> to continue.");
    _ = Console.ReadLine();
    Console.WriteLine("\n");
}

// Returns the string textToCenter, padded on the left with spaces
// that center the text on the console display.
```

```
static string CenterText(string textToCenter)
{
    var centeredText = new StringBuilder();
    var screenWidth = Console.WindowWidth;
    centeredText.Append(new string(' ', (int)(screenWidth -
 textToCenter.Length) / 2));
    centeredText.Append(textToCenter);
    return centeredText.ToString();
}

// Displays a list of file names included in the specified path.
static void DisplayLocalFiles(string localPath)
{
    var fileList = Directory.GetFiles(localPath);
    if (fileList.Length > 0)
    {
        foreach (var fileName in fileList)
        {
            Console.WriteLine(fileName);
        }
    }
}

// Displays a list of the files in the specified S3 bucket and prefix.
static async Task DisplayBucketFiles(IAmazonS3 client, string bucketName, string
 s3Path)
{
    ListObjectsV2Request request = new()
    {
        BucketName = bucketName,
        Prefix = s3Path,
        MaxKeys = 5,
    };

    var response = new ListObjectsV2Response();

    do
    {
        response = await client.ListObjectsV2Async(request);

        response.S3Objects
            .ForEach(obj => Console.WriteLine($"{obj.Key}"));

        // If the response is truncated, set the request ContinuationToken
```

```
        // from the NextContinuationToken property of the response.
        request.ContinuationToken = response.NextContinuationToken;
    } while (response.IsTruncated);
}
```

Unggah file tunggal.

```
        /// <summary>
        /// Uploads a single file from the local computer to an S3 bucket.
        /// </summary>
        /// <param name="transferUtil">The transfer initialized TransferUtility
        /// object.</param>
        /// <param name="bucketName">The name of the S3 bucket where the file
        /// will be stored.</param>
        /// <param name="fileName">The name of the file to upload.</param>
        /// <param name="localPath">The local path where the file is stored.</
param>
        /// <returns>A boolean value indicating the success of the action.</
returns>
        public static async Task<bool> UploadSingleFileAsync(
            TransferUtility transferUtil,
            string bucketName,
            string fileName,
            string localPath)
        {
            if (File.Exists($"{localPath}\\{fileName}"))
            {
                try
                {
                    await transferUtil.UploadAsync(new
 TransferUtilityUploadRequest
                    {
                        BucketName = bucketName,
                        Key = fileName,
                        FilePath = $"{localPath}\\{fileName}",
                    });

                    return true;
                }
                catch (AmazonS3Exception s3Ex)
```

```
                {
                    Console.WriteLine($"Could not upload {fileName} from
 {localPath} because:");
                    Console.WriteLine(s3Ex.Message);
                    return false;
                }
            }
            else
            {
                Console.WriteLine($"{fileName} does not exist in {localPath}");
                return false;
            }
        }
```

Unggah seluruh direktori lokal.

```
        /// <summary>
        /// Uploads all the files in a local directory to a directory in an S3
        /// bucket.
        /// </summary>
        /// <param name="transferUtil">The transfer initialized TransferUtility
        /// object.</param>
        /// <param name="bucketName">The name of the S3 bucket where the files
        /// will be stored.</param>
        /// <param name="keyPrefix">The key prefix is the S3 directory where
        /// the files will be stored.</param>
        /// <param name="localPath">The local directory that contains the files
        /// to be uploaded.</param>
        /// <returns>A Boolean value representing the success of the action.</
returns>
        public static async Task<bool> UploadFullDirectoryAsync(
            TransferUtility transferUtil,
            string bucketName,
            string keyPrefix,
            string localPath)
        {
            if (Directory.Exists(localPath))
            {
                try
                {
```

```
                    await transferUtil.UploadDirectoryAsync(new
 TransferUtilityUploadDirectoryRequest
                    {
                        BucketName = bucketName,
                        KeyPrefix = keyPrefix,
                        Directory = localPath,
                    });

                    return true;
                }
                catch (AmazonS3Exception s3Ex)
                {
                    Console.WriteLine($"Can't upload the contents of {localPath}
 because:");
                    Console.WriteLine(s3Ex?.Message);
                    return false;
                }
            }
            else
            {
                Console.WriteLine($"The directory {localPath} does not exist.");
                return false;
            }
        }
```

Unduh file tunggal.

```
        /// <summary>
        /// Download a single file from an S3 bucket to the local computer.
        /// </summary>
        /// <param name="transferUtil">The transfer initialized TransferUtility
        /// object.</param>
        /// <param name="bucketName">The name of the S3 bucket containing the
        /// file to download.</param>
        /// <param name="keyName">The name of the file to download.</param>
        /// <param name="localPath">The path on the local computer where the
        /// downloaded file will be saved.</param>
        /// <returns>A Boolean value indicating the results of the action.</
returns>
        public static async Task<bool> DownloadSingleFileAsync(
```

```
        TransferUtility transferUtil,
            string bucketName,
            string keyName,
            string localPath)
        {
            await transferUtil.DownloadAsync(new TransferUtilityDownloadRequest
            {
                BucketName = bucketName,
                Key = keyName,
                FilePath = $"{localPath}\\{keyName}",
            });

            return (File.Exists($"{localPath}\\{keyName}"));
        }
```

Unduh konten bucket S3.

```
        /// <summary>
        /// Downloads the contents of a directory in an S3 bucket to a
        /// directory on the local computer.
        /// </summary>
        /// <param name="transferUtil">The transfer initialized TransferUtility
        /// object.</param>
        /// <param name="bucketName">The bucket containing the files to
 download.</param>
        /// <param name="s3Path">The S3 directory where the files are located.</
param>
        /// <param name="localPath">The local path to which the files will be
        /// saved.</param>
        /// <returns>A Boolean value representing the success of the action.</
returns>
        public static async Task<bool> DownloadS3DirectoryAsync(
            TransferUtility transferUtil,
            string bucketName,
            string s3Path,
            string localPath)
        {
            int fileCount = 0;

            // If the directory doesn't exist, it will be created.
```

```
            if (Directory.Exists(s3Path))
            {
                var files = Directory.GetFiles(localPath);
                fileCount = files.Length;
            }

            await transferUtil.DownloadDirectoryAsync(new
 TransferUtilityDownloadDirectoryRequest
            {
                BucketName = bucketName,
                LocalDirectory = localPath,
                S3Directory = s3Path,
            });

            if (Directory.Exists(localPath))
            {
                var files = Directory.GetFiles(localPath);
                if (files.Length > fileCount)
                {
                    return true;
                }

                // No change in the number of files. Assume
                // the download failed.
                return false;
            }

            // The local directory doesn't exist. No files
            // were downloaded.
            return false;
        }
```

Lacak kemajuan unggahan menggunakan file TransferUtility.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.S3;
    using Amazon.S3.Transfer;

    /// <summary>
    /// This example shows how to track the progress of a multipart upload
```

```
    /// using the Amazon Simple Storage Service (Amazon S3) TransferUtility to
    /// upload to an Amazon S3 bucket.
    /// </summary>
    public class TrackMPUUsingHighLevelAPI
    {
        public static async Task Main()
        {
            string bucketName = "doc-example-bucket";
            string keyName = "sample_pic.png";
            string path = "filepath/directory/";
            string filePath = $"{path}{keyName}";

            // If the AWS Region defined for your default user is different
            // from the Region where your Amazon S3 bucket is located,
            // pass the Region name to the Amazon S3 client object's constructor.
            // For example: RegionEndpoint.USWest2 or RegionEndpoint.USEast2.
            IAmazonS3 client = new AmazonS3Client();

            await TrackMPUAsync(client, bucketName, filePath, keyName);
        }

        /// <summary>
        /// Starts an Amazon S3 multipart upload and assigns an event handler to
        /// track the progress of the upload.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 client object used to
        /// perform the multipart upload.</param>
        /// <param name="bucketName">The name of the bucket to which to upload
        /// the file.</param>
        /// <param name="filePath">The path, including the file name of the
        /// file to be uploaded to the Amazon S3 bucket.</param>
        /// <param name="keyName">The file name to be used in the
        /// destination Amazon S3 bucket.</param>
        public static async Task TrackMPUAsync(
            IAmazonS3 client,
            string bucketName,
            string filePath,
            string keyName)
        {
            try
            {
                var fileTransferUtility = new TransferUtility(client);

                // Use TransferUtilityUploadRequest to configure options.
```

```
                // In this example we subscribe to an event.
                var uploadRequest =
                    new TransferUtilityUploadRequest
                    {
                        BucketName = bucketName,
                        FilePath = filePath,
                        Key = keyName,
                    };

                uploadRequest.UploadProgressEvent +=
                    new EventHandler<UploadProgressArgs>(
                        UploadRequest_UploadPartProgressEvent);

                await fileTransferUtility.UploadAsync(uploadRequest);
                Console.WriteLine("Upload completed");
            }
            catch (AmazonS3Exception ex)
            {
                Console.WriteLine($"Error:: {ex.Message}");
            }
        }

        /// <summary>
        /// Event handler to check the progress of the multipart upload.
        /// </summary>
        /// <param name="sender">The object that raised the event.</param>
        /// <param name="e">The object that contains multipart upload
        /// information.</param>
        public static void UploadRequest_UploadPartProgressEvent(object sender,
UploadProgressArgs e)
        {
            // Process event.
            Console.WriteLine($"{e.TransferredBytes}/{e.TotalBytes}");
        }
    }
```

Unggah objek dengan enkripsi.

```
    using System;
    using System.Collections.Generic;
    using System.IO;
```

```
using System.Security.Cryptography;
using System.Threading.Tasks;
using Amazon.S3;
using Amazon.S3.Model;

/// <summary>
/// Uses the Amazon Simple Storage Service (Amazon S3) low level API to
/// perform a multipart upload to an Amazon S3 bucket.
/// </summary>
public class SSECLowLevelMPUcopyObject
{
    public static async Task Main()
    {
        string existingBucketName = "doc-example-bucket";
        string sourceKeyName = "sample_file.txt";
        string targetKeyName = "sample_file_copy.txt";
        string filePath = $"sample\\{targetKeyName}";

        // If the AWS Region defined for your default user is different
        // from the Region where your Amazon S3 bucket is located,
        // pass the Region name to the Amazon S3 client object's constructor.
        // For example: RegionEndpoint.USEast1.
        IAmazonS3 client = new AmazonS3Client();

        // Create the encryption key.
        var base64Key = CreateEncryptionKey();

        await CreateSampleObjUsingClientEncryptionKeyAsync(
            client,
            existingBucketName,
            sourceKeyName,
            filePath,
            base64Key);
    }

    /// <summary>
    /// Creates the encryption key to use with the multipart upload.
    /// </summary>
    /// <returns>A string containing the base64-encoded key for encrypting
    /// the multipart upload.</returns>
    public static string CreateEncryptionKey()
    {
        Aes aesEncryption = Aes.Create();
        aesEncryption.KeySize = 256;
```

```
            aesEncryption.GenerateKey();
            string base64Key = Convert.ToBase64String(aesEncryption.Key);
            return base64Key;
        }

        /// <summary>
        /// Creates and uploads an object using a multipart upload.
        /// </summary>
        /// <param name="client">The initialized Amazon S3 object used to
        /// initialize and perform the multipart upload.</param>
        /// <param name="existingBucketName">The name of the bucket to which
        /// the object will be uploaded.</param>
        /// <param name="sourceKeyName">The source object name.</param>
        /// <param name="filePath">The location of the source object.</param>
        /// <param name="base64Key">The encryption key to use with the upload.</
param>
        public static async Task CreateSampleObjUsingClientEncryptionKeyAsync(
            IAmazonS3 client,
            string existingBucketName,
            string sourceKeyName,
            string filePath,
            string base64Key)
        {
            List<UploadPartResponse> uploadResponses = new
 List<UploadPartResponse>();

            InitiateMultipartUploadRequest initiateRequest = new
 InitiateMultipartUploadRequest
            {
                BucketName = existingBucketName,
                Key = sourceKeyName,
                ServerSideEncryptionCustomerMethod =
 ServerSideEncryptionCustomerMethod.AES256,
                ServerSideEncryptionCustomerProvidedKey = base64Key,
            };

            InitiateMultipartUploadResponse initResponse =
               await client.InitiateMultipartUploadAsync(initiateRequest);

            long contentLength = new FileInfo(filePath).Length;
            long partSize = 5 * (long)Math.Pow(2, 20); // 5 MB

            try
            {
```

```
                long filePosition = 0;
                for (int i = 1; filePosition < contentLength; i++)
                {
                    UploadPartRequest uploadRequest = new UploadPartRequest
                    {
                        BucketName = existingBucketName,
                        Key = sourceKeyName,
                        UploadId = initResponse.UploadId,
                        PartNumber = i,
                        PartSize = partSize,
                        FilePosition = filePosition,
                        FilePath = filePath,
                        ServerSideEncryptionCustomerMethod =
ServerSideEncryptionCustomerMethod.AES256,
                        ServerSideEncryptionCustomerProvidedKey = base64Key,
                    };

                    // Upload part and add response to our list.
                    uploadResponses.Add(await
client.UploadPartAsync(uploadRequest));

                    filePosition += partSize;
                }

                CompleteMultipartUploadRequest completeRequest = new
CompleteMultipartUploadRequest
                {
                    BucketName = existingBucketName,
                    Key = sourceKeyName,
                    UploadId = initResponse.UploadId,
                };
                completeRequest.AddPartETags(uploadResponses);

                CompleteMultipartUploadResponse completeUploadResponse =
                    await client.CompleteMultipartUploadAsync(completeRequest);
            }
            catch (Exception exception)
            {
                Console.WriteLine($"Exception occurred: {exception.Message}");

                // If there was an error, abort the multipart upload.
                AbortMultipartUploadRequest abortMPURequest = new
AbortMultipartUploadRequest
                {
```

```
                    BucketName = existingBucketName,
                    Key = sourceKeyName,
                    UploadId = initResponse.UploadId,
                };

                await client.AbortMultipartUploadAsync(abortMPURequest);
            }
        }
    }
```

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Unggah objek besar dengan menggunakan pengelola unggahan untuk memecah data menjadi beberapa bagian dan mengunggahnya secara bersamaan.

```
// BucketBasics encapsulates the Amazon Simple Storage Service (Amazon S3)
 actions
// used in the examples.
// It contains S3Client, an Amazon S3 service client that is used to perform
 bucket
// and object actions.
type BucketBasics struct {
 S3Client *s3.Client
}



// UploadLargeObject uses an upload manager to upload data to an object in a
 bucket.
```

```
// The upload manager breaks large data into parts and uploads the parts
 concurrently.
func (basics BucketBasics) UploadLargeObject(bucketName string, objectKey string,
 largeObject []byte) error {
 largeBuffer := bytes.NewReader(largeObject)
 var partMiBs int64 = 10
 uploader := manager.NewUploader(basics.S3Client, func(u *manager.Uploader) {
  u.PartSize = partMiBs * 1024 * 1024
 })
 _, err := uploader.Upload(context.TODO(), &s3.PutObjectInput{
  Bucket: aws.String(bucketName),
  Key:    aws.String(objectKey),
  Body:   largeBuffer,
 })
 if err != nil {
  log.Printf("Couldn't upload large object to %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }

 return err
}
```

Unduh objek besar dengan menggunakan pengelola unduhan untuk mendapatkan data dalam beberapa bagian dan mengunduhnya secara bersamaan.

```
// DownloadLargeObject uses a download manager to download an object from a
 bucket.
// The download manager gets the data in parts and writes them to a buffer until
 all of
// the data has been downloaded.
func (basics BucketBasics) DownloadLargeObject(bucketName string, objectKey
 string) ([]byte, error) {
 var partMiBs int64 = 10
 downloader := manager.NewDownloader(basics.S3Client, func(d *manager.Downloader)
 {
  d.PartSize = partMiBs * 1024 * 1024
 })
 buffer := manager.NewWriteAtBuffer([]byte{})
 _, err := downloader.Download(context.TODO(), buffer, &s3.GetObjectInput{
  Bucket: aws.String(bucketName),
```

```
  Key:    aws.String(objectKey),
 })
 if err != nil {
  log.Printf("Couldn't download large object from %v:%v. Here's why: %v\n",
   bucketName, objectKey, err)
 }
 return buffer.Bytes(), err
}
```

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Fungsi panggilan yang mentransfer file ke dan dari bucket S3 menggunakan TransferManager S3.

```
    public Integer downloadObjectsToDirectory(S3TransferManager transferManager,
            URI destinationPathURI, String bucketName) {
        DirectoryDownload directoryDownload =
transferManager.downloadDirectory(DownloadDirectoryRequest.builder()
                .destination(Paths.get(destinationPathURI))
                .bucket(bucketName)
                .build());
        CompletedDirectoryDownload completedDirectoryDownload =
directoryDownload.completionFuture().join();

        completedDirectoryDownload.failedTransfers()
                .forEach(fail -> logger.warn("Object [{}] failed to transfer",
fail.toString()));
        return completedDirectoryDownload.failedTransfers().size();
    }
```

Unggah seluruh direktori lokal.

```
   public Integer uploadDirectory(S3TransferManager transferManager,
           URI sourceDirectory, String bucketName) {
       DirectoryUpload directoryUpload =
transferManager.uploadDirectory(UploadDirectoryRequest.builder()
               .source(Paths.get(sourceDirectory))
               .bucket(bucketName)
               .build());

       CompletedDirectoryUpload completedDirectoryUpload =
directoryUpload.completionFuture().join();
       completedDirectoryUpload.failedTransfers()
               .forEach(fail -> logger.warn("Object [{}] failed to transfer",
fail.toString()));
       return completedDirectoryUpload.failedTransfers().size();
   }
```

Unggah file tunggal.

```
   public String uploadFile(S3TransferManager transferManager, String
bucketName,
                            String key, URI filePathURI) {
       UploadFileRequest uploadFileRequest = UploadFileRequest.builder()
           .putObjectRequest(b -> b.bucket(bucketName).key(key))
           .source(Paths.get(filePathURI))
           .build();

       FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);

       CompletedFileUpload uploadResult = fileUpload.completionFuture().join();
       return uploadResult.response().eTag();
   }
```

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Unggah file besar.

```
import {
  CreateMultipartUploadCommand,
  UploadPartCommand,
  CompleteMultipartUploadCommand,
  AbortMultipartUploadCommand,
  S3Client,
} from "@aws-sdk/client-s3";

const twentyFiveMB = 25 * 1024 * 1024;

export const createString = (size = twentyFiveMB) => {
  return "x".repeat(size);
};

export const main = async () => {
  const s3Client = new S3Client({});
  const bucketName = "test-bucket";
  const key = "multipart.txt";
  const str = createString();
  const buffer = Buffer.from(str, "utf8");

  let uploadId;

  try {
    const multipartUpload = await s3Client.send(
      new CreateMultipartUploadCommand({
        Bucket: bucketName,
        Key: key,
      }),
    );
```

```
    uploadId = multipartUpload.UploadId;

    const uploadPromises = [];
    // Multipart uploads require a minimum size of 5 MB per part.
    const partSize = Math.ceil(buffer.length / 5);

    // Upload each part.
    for (let i = 0; i < 5; i++) {
      const start = i * partSize;
      const end = start + partSize;
      uploadPromises.push(
        s3Client
          .send(
            new UploadPartCommand({
              Bucket: bucketName,
              Key: key,
              UploadId: uploadId,
              Body: buffer.subarray(start, end),
              PartNumber: i + 1,
            }),
          )
          .then((d) => {
            console.log("Part", i + 1, "uploaded");
            return d;
          }),
      );
    }

    const uploadResults = await Promise.all(uploadPromises);

    return await s3Client.send(
      new CompleteMultipartUploadCommand({
        Bucket: bucketName,
        Key: key,
        UploadId: uploadId,
        MultipartUpload: {
          Parts: uploadResults.map(({ ETag }, i) => ({
            ETag,
            PartNumber: i + 1,
          })),
        },
      }),
    );
```

```
    // Verify the output by downloading the file from the Amazon Simple Storage
 Service (Amazon S3) console.
    // Because the output is a 25 MB string, text editors might struggle to open
 the file.
  } catch (err) {
    console.error(err);

    if (uploadId) {
      const abortCommand = new AbortMultipartUploadCommand({
        Bucket: bucketName,
        Key: key,
        UploadId: uploadId,
      });

      await s3Client.send(abortCommand);
    }
  }
};
```

Unduh file besar.

```
import { GetObjectCommand, S3Client } from "@aws-sdk/client-s3";
import { createWriteStream } from "fs";

const s3Client = new S3Client({});
const oneMB = 1024 * 1024;

export const getObjectRange = ({ bucket, key, start, end }) => {
  const command = new GetObjectCommand({
    Bucket: bucket,
    Key: key,
    Range: `bytes=${start}-${end}`,
  });

  return s3Client.send(command);
};

/**
 * @param {string | undefined} contentRange
 */
export const getRangeAndLength = (contentRange) => {
  const [range, length] = contentRange.split("/");
```

```
  const [start, end] = range.split("-");
  return {
    start: parseInt(start),
    end: parseInt(end),
    length: parseInt(length),
  };
};

export const isComplete = ({ end, length }) => end === length - 1;

// When downloading a large file, you might want to break it down into
// smaller pieces. Amazon S3 accepts a Range header to specify the start
// and end of the byte range to be downloaded.
const downloadInChunks = async ({ bucket, key }) => {
  const writeStream = createWriteStream(
    fileURLToPath(new URL(`./${key}`, import.meta.url)),
  ).on("error", (err) => console.error(err));

  let rangeAndLength = { start: -1, end: -1, length: -1 };

  while (!isComplete(rangeAndLength)) {
    const { end } = rangeAndLength;
    const nextRange = { start: end + 1, end: end + oneMB };

    console.log(`Downloading bytes ${nextRange.start} to ${nextRange.end}`);

    const { ContentRange, Body } = await getObjectRange({
      bucket,
      key,
      ...nextRange,
    });

    writeStream.write(await Body.transformToByteArray());
    rangeAndLength = getRangeAndLength(ContentRange);
  }
};

export const main = async () => {
  await downloadInChunks({
    bucket: "my-cool-bucket",
    key: "my-cool-object.txt",
  });
};
```

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat fungsi yang mentransfer file menggunakan beberapa pengaturan manajer transfer yang tersedia. Gunakan kelas panggilan balik untuk menulis progres callback selama transfer file.

```
import sys
import threading

import boto3
from boto3.s3.transfer import TransferConfig


MB = 1024 * 1024
s3 = boto3.resource("s3")


class TransferCallback:
    """
    Handle callbacks from the transfer manager.

    The transfer manager periodically calls the __call__ method throughout
    the upload and download process so that it can take action, such as
    displaying progress to the user and collecting data about the transfer.
    """

    def __init__(self, target_size):
        self._target_size = target_size
        self._total_transferred = 0
        self._lock = threading.Lock()
        self.thread_info = {}
```

```
    def __call__(self, bytes_transferred):
        """
        The callback method that is called by the transfer manager.

        Display progress during file transfer and collect per-thread transfer
        data. This method can be called by multiple threads, so shared instance
        data is protected by a thread lock.
        """
        thread = threading.current_thread()
        with self._lock:
            self._total_transferred += bytes_transferred
            if thread.ident not in self.thread_info.keys():
                self.thread_info[thread.ident] = bytes_transferred
            else:
                self.thread_info[thread.ident] += bytes_transferred

            target = self._target_size * MB
            sys.stdout.write(
                f"\r{self._total_transferred} of {target} transferred "
                f"({(self._total_transferred / target) * 100:.2f}%)."
            )
            sys.stdout.flush()


def upload_with_default_configuration(
    local_file_path, bucket_name, object_key, file_size_mb
):
    """
    Upload a file from a local folder to an Amazon S3 bucket, using the default
    configuration.
    """
    transfer_callback = TransferCallback(file_size_mb)
    s3.Bucket(bucket_name).upload_file(
        local_file_path, object_key, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def upload_with_chunksize_and_meta(
    local_file_path, bucket_name, object_key, file_size_mb, metadata=None
):
    """
    Upload a file from a local folder to an Amazon S3 bucket, setting a
    multipart chunk size and adding metadata to the Amazon S3 object.
```

```
    The multipart chunk size controls the size of the chunks of data that are
    sent in the request. A smaller chunk size typically results in the transfer
    manager using more threads for the upload.

    The metadata is a set of key-value pairs that are stored with the object
    in Amazon S3.
    """
    transfer_callback = TransferCallback(file_size_mb)

    config = TransferConfig(multipart_chunksize=1 * MB)
    extra_args = {"Metadata": metadata} if metadata else None
    s3.Bucket(bucket_name).upload_file(
        local_file_path,
        object_key,
        Config=config,
        ExtraArgs=extra_args,
        Callback=transfer_callback,
    )
    return transfer_callback.thread_info


def upload_with_high_threshold(local_file_path, bucket_name, object_key,
 file_size_mb):
    """
    Upload a file from a local folder to an Amazon S3 bucket, setting a
    multipart threshold larger than the size of the file.

    Setting a multipart threshold larger than the size of the file results
    in the transfer manager sending the file as a standard upload instead of
    a multipart upload.
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(multipart_threshold=file_size_mb * 2 * MB)
    s3.Bucket(bucket_name).upload_file(
        local_file_path, object_key, Config=config, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def upload_with_sse(
    local_file_path, bucket_name, object_key, file_size_mb, sse_key=None
):
    """
```

```
    Upload a file from a local folder to an Amazon S3 bucket, adding server-side
    encryption with customer-provided encryption keys to the object.

    When this kind of encryption is specified, Amazon S3 encrypts the object
    at rest and allows downloads only when the expected encryption key is
    provided in the download request.
    """
    transfer_callback = TransferCallback(file_size_mb)
    if sse_key:
        extra_args = {"SSECustomerAlgorithm": "AES256", "SSECustomerKey":
 sse_key}
    else:
        extra_args = None
    s3.Bucket(bucket_name).upload_file(
        local_file_path, object_key, ExtraArgs=extra_args,
 Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_default_configuration(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, using the
    default configuration.
    """
    transfer_callback = TransferCallback(file_size_mb)
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_single_thread(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, using a
    single thread.
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(use_threads=False)
    s3.Bucket(bucket_name).Object(object_key).download_file(
```

```
        download_file_path, Config=config, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_high_threshold(
    bucket_name, object_key, download_file_path, file_size_mb
):
    """
    Download a file from an Amazon S3 bucket to a local folder, setting a
    multipart threshold larger than the size of the file.

    Setting a multipart threshold larger than the size of the file results
    in the transfer manager sending the file as a standard download instead
    of a multipart download.
    """
    transfer_callback = TransferCallback(file_size_mb)
    config = TransferConfig(multipart_threshold=file_size_mb * 2 * MB)
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, Config=config, Callback=transfer_callback
    )
    return transfer_callback.thread_info


def download_with_sse(
    bucket_name, object_key, download_file_path, file_size_mb, sse_key
):
    """
    Download a file from an Amazon S3 bucket to a local folder, adding a
    customer-provided encryption key to the request.

    When this kind of encryption is specified, Amazon S3 encrypts the object
    at rest and allows downloads only when the expected encryption key is
    provided in the download request.
    """
    transfer_callback = TransferCallback(file_size_mb)

    if sse_key:
        extra_args = {"SSECustomerAlgorithm": "AES256", "SSECustomerKey":
 sse_key}
    else:
        extra_args = None
    s3.Bucket(bucket_name).Object(object_key).download_file(
        download_file_path, ExtraArgs=extra_args, Callback=transfer_callback
```

```
    )
    return transfer_callback.thread_info
```

Menunjukkan fungsi manajer transfer dan melaporkan hasil.

```
import hashlib
import os
import platform
import shutil
import time

import boto3
from boto3.s3.transfer import TransferConfig
from botocore.exceptions import ClientError
from botocore.exceptions import ParamValidationError
from botocore.exceptions import NoCredentialsError

import file_transfer

MB = 1024 * 1024
# These configuration attributes affect both uploads and downloads.
CONFIG_ATTRS = (
    "multipart_threshold",
    "multipart_chunksize",
    "max_concurrency",
    "use_threads",
)
# These configuration attributes affect only downloads.
DOWNLOAD_CONFIG_ATTRS = ("max_io_queue", "io_chunksize", "num_download_attempts")


class TransferDemoManager:
    """
    Manages the demonstration. Collects user input from a command line, reports
    transfer results, maintains a list of artifacts created during the
    demonstration, and cleans them up after the demonstration is completed.
    """

    def __init__(self):
        self._s3 = boto3.resource("s3")
```

```
        self._chore_list = []
        self._create_file_cmd = None
        self._size_multiplier = 0
        self.file_size_mb = 30
        self.demo_folder = None
        self.demo_bucket = None
        self._setup_platform_specific()
        self._terminal_width = shutil.get_terminal_size(fallback=(80, 80))[0]

    def collect_user_info(self):
        """
        Collect local folder and Amazon S3 bucket name from the user. These
        locations are used to store files during the demonstration.
        """
        while not self.demo_folder:
            self.demo_folder = input(
                "Which file folder do you want to use to store " "demonstration
files? "
            )
            if not os.path.isdir(self.demo_folder):
                print(f"{self.demo_folder} isn't a folder!")
                self.demo_folder = None

        while not self.demo_bucket:
            self.demo_bucket = input(
                "Which Amazon S3 bucket do you want to use to store "
                "demonstration files? "
            )
            try:
                self._s3.meta.client.head_bucket(Bucket=self.demo_bucket)
            except ParamValidationError as err:
                print(err)
                self.demo_bucket = None
            except ClientError as err:
                print(err)
                print(
                    f"Either {self.demo_bucket} doesn't exist or you don't "
                    f"have access to it."
                )
                self.demo_bucket = None

    def demo(
        self, question, upload_func, download_func, upload_args=None,
download_args=None
```

```
    ):
        """Run a demonstration.

        Ask the user if they want to run this specific demonstration.
        If they say yes, create a file on the local path, upload it
        using the specified upload function, then download it using the
        specified download function.
        """
        if download_args is None:
            download_args = {}
        if upload_args is None:
            upload_args = {}
        question = question.format(self.file_size_mb)
        answer = input(f"{question} (y/n)")
        if answer.lower() == "y":
            local_file_path, object_key, download_file_path =
self._create_demo_file()

            file_transfer.TransferConfig = self._config_wrapper(
                TransferConfig, CONFIG_ATTRS
            )
            self._report_transfer_params(
                "Uploading", local_file_path, object_key, **upload_args
            )
            start_time = time.perf_counter()
            thread_info = upload_func(
                local_file_path,
                self.demo_bucket,
                object_key,
                self.file_size_mb,
                **upload_args,
            )
            end_time = time.perf_counter()
            self._report_transfer_result(thread_info, end_time - start_time)

            file_transfer.TransferConfig = self._config_wrapper(
                TransferConfig, CONFIG_ATTRS + DOWNLOAD_CONFIG_ATTRS
            )
            self._report_transfer_params(
                "Downloading", object_key, download_file_path, **download_args
            )
            start_time = time.perf_counter()
            thread_info = download_func(
                self.demo_bucket,
```

```
                object_key,
                download_file_path,
                self.file_size_mb,
                **download_args,
            )
            end_time = time.perf_counter()
            self._report_transfer_result(thread_info, end_time - start_time)

    def last_name_set(self):
        """Get the name set used for the last demo."""
        return self._chore_list[-1]

    def cleanup(self):
        """
        Remove files from the demo folder, and uploaded objects from the
        Amazon S3 bucket.
        """
        print("-" * self._terminal_width)
        for local_file_path, s3_object_key, downloaded_file_path in
self._chore_list:
            print(f"Removing {local_file_path}")
            try:
                os.remove(local_file_path)
            except FileNotFoundError as err:
                print(err)

            print(f"Removing {downloaded_file_path}")
            try:
                os.remove(downloaded_file_path)
            except FileNotFoundError as err:
                print(err)

            if self.demo_bucket:
                print(f"Removing {self.demo_bucket}:{s3_object_key}")
                try:

self._s3.Bucket(self.demo_bucket).Object(s3_object_key).delete()
                except ClientError as err:
                    print(err)

    def _setup_platform_specific(self):
        """Set up platform-specific command used to create a large file."""
        if platform.system() == "Windows":
            self._create_file_cmd = "fsutil file createnew {} {}"
```

```
            self._size_multiplier = MB
        elif platform.system() == "Linux" or platform.system() == "Darwin":
            self._create_file_cmd = f"dd if=/dev/urandom of={{}} " f"bs={MB}
count={{}}"
            self._size_multiplier = 1
        else:
            raise EnvironmentError(
                f"Demo of platform {platform.system()} isn't supported."
            )

    def _create_demo_file(self):
        """
        Create a file in the demo folder specified by the user. Store the local
        path, object name, and download path for later cleanup.

        Only the local file is created by this method. The Amazon S3 object and
        download file are created later during the demonstration.

        Returns:
        A tuple that contains the local file path, object name, and download
        file path.
        """
        file_name_template = "TestFile{}-{}.demo"
        local_suffix = "local"
        object_suffix = "s3object"
        download_suffix = "downloaded"
        file_tag = len(self._chore_list) + 1

        local_file_path = os.path.join(
            self.demo_folder, file_name_template.format(file_tag, local_suffix)
        )

        s3_object_key = file_name_template.format(file_tag, object_suffix)

        downloaded_file_path = os.path.join(
            self.demo_folder, file_name_template.format(file_tag,
download_suffix)
        )

        filled_cmd = self._create_file_cmd.format(
            local_file_path, self.file_size_mb * self._size_multiplier
        )

        print(
```

```
            f"Creating file of size {self.file_size_mb} MB "
            f"in {self.demo_folder} by running:"
        )
        print(f"{'':4}{filled_cmd}")
        os.system(filled_cmd)

        chore = (local_file_path, s3_object_key, downloaded_file_path)
        self._chore_list.append(chore)
        return chore

    def _report_transfer_params(self, verb, source_name, dest_name, **kwargs):
        """Report configuration and extra arguments used for a file transfer."""
        print("-" * self._terminal_width)
        print(f"{verb} {source_name} ({self.file_size_mb} MB) to {dest_name}")
        if kwargs:
            print("With extra args:")
            for arg, value in kwargs.items():
                print(f'{"":4}{arg:<20}: {value}')

    @staticmethod
    def ask_user(question):
        """
        Ask the user a yes or no question.

        Returns:
        True when the user answers 'y' or 'Y'; otherwise, False.
        """
        answer = input(f"{question} (y/n) ")
        return answer.lower() == "y"

    @staticmethod
    def _config_wrapper(func, config_attrs):
        def wrapper(*args, **kwargs):
            config = func(*args, **kwargs)
            print("With configuration:")
            for attr in config_attrs:
                print(f'{"":4}{attr:<20}: {getattr(config, attr)}')
            return config

        return wrapper

    @staticmethod
    def _report_transfer_result(thread_info, elapsed):
        """Report the result of a transfer, including per-thread data."""
```

```
        print(f"\nUsed {len(thread_info)} threads.")
        for ident, byte_count in thread_info.items():
            print(f"{'':4}Thread {ident} copied {byte_count} bytes.")
        print(f"Your transfer took {elapsed:.2f} seconds.")


def main():
    """
    Run the demonstration script for s3_file_transfer.
    """
    demo_manager = TransferDemoManager()
    demo_manager.collect_user_info()

    # Upload and download with default configuration. Because the file is 30 MB
    # and the default multipart_threshold is 8 MB, both upload and download are
    # multipart transfers.
    demo_manager.demo(
        "Do you want to upload and download a {} MB file "
        "using the default configuration?",
        file_transfer.upload_with_default_configuration,
        file_transfer.download_with_default_configuration,
    )

    # Upload and download with multipart_threshold set higher than the size of
    # the file. This causes the transfer manager to use standard transfers
    # instead of multipart transfers.
    demo_manager.demo(
        "Do you want to upload and download a {} MB file "
        "as a standard (not multipart) transfer?",
        file_transfer.upload_with_high_threshold,
        file_transfer.download_with_high_threshold,
    )

    # Upload with specific chunk size and additional metadata.
    # Download with a single thread.
    demo_manager.demo(
        "Do you want to upload a {} MB file with a smaller chunk size and "
        "then download the same file using a single thread?",
        file_transfer.upload_with_chunksize_and_meta,
        file_transfer.download_with_single_thread,
        upload_args={
            "metadata": {
                "upload_type": "chunky",
                "favorite_color": "aqua",
```

```
                "size": "medium",
            }
        },
    )

    # Upload using server-side encryption with customer-provided
    # encryption keys.
    # Generate a 256-bit key from a passphrase.
    sse_key = hashlib.sha256("demo_passphrase".encode("utf-8")).digest()
    demo_manager.demo(
        "Do you want to upload and download a {} MB file using "
        "server-side encryption?",
        file_transfer.upload_with_sse,
        file_transfer.download_with_sse,
        upload_args={"sse_key": sse_key},
        download_args={"sse_key": sse_key},
    )

    # Download without specifying an encryption key to show that the
    # encryption key must be included to download an encrypted object.
    if demo_manager.ask_user(
        "Do you want to try to download the encrypted "
        "object without sending the required key?"
    ):
        try:
            _, object_key, download_file_path = demo_manager.last_name_set()
            file_transfer.download_with_default_configuration(
                demo_manager.demo_bucket,
                object_key,
                download_file_path,
                demo_manager.file_size_mb,
            )
        except ClientError as err:
            print(
                "Got expected error when trying to download an encrypted "
                "object without specifying encryption info:"
            )
            print(f"{'':4}{err}")

    # Remove all created and downloaded files, remove all objects from
    # S3 storage.
    if demo_manager.ask_user(
        "Demonstration complete. Do you want to remove local files " "and S3
objects?"
```

```
    ):
        demo_manager.cleanup()


if __name__ == "__main__":
    try:
        main()
    except NoCredentialsError as error:
        print(error)
        print(
            "To run this example, you must have valid credentials in "
            "a shared credential file or set in environment variables."
        )
```

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
use std::fs::File;
use std::io::prelude::*;
use std::path::Path;

use aws_config::meta::region::RegionProviderChain;
use aws_sdk_s3::error::DisplayErrorContext;
use aws_sdk_s3::operation::{
    create_multipart_upload::CreateMultipartUploadOutput,
 get_object::GetObjectOutput,
};
use aws_sdk_s3::types::{CompletedMultipartUpload, CompletedPart};
use aws_sdk_s3::{config::Region, Client as S3Client};
use aws_smithy_types::byte_stream::{ByteStream, Length};
use rand::distributions::Alphanumeric;
use rand::{thread_rng, Rng};
```

```
use s3_service::error::Error;
use std::process;
use uuid::Uuid;

//In bytes, minimum chunk size of 5MB. Increase CHUNK_SIZE to send larger chunks.
const CHUNK_SIZE: u64 = 1024 * 1024 * 5;
const MAX_CHUNKS: u64 = 10000;

#[tokio::main]
pub async fn main() {
    if let Err(err) = run_example().await {
        eprintln!("Error: {}", DisplayErrorContext(err));
        process::exit(1);
    }
}

async fn run_example() -> Result<(), Error> {
    let shared_config = aws_config::load_from_env().await;
    let client = S3Client::new(&shared_config);

    let bucket_name = format!("doc-example-bucket-{}", Uuid::new_v4());
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));
    let region = region_provider.region().await.unwrap();
    s3_service::create_bucket(&client, &bucket_name, region.as_ref()).await?;

    let key = "sample.txt".to_string();
    let multipart_upload_res: CreateMultipartUploadOutput = client
        .create_multipart_upload()
        .bucket(&bucket_name)
        .key(&key)
        .send()
        .await
        .unwrap();
    let upload_id = multipart_upload_res.upload_id().unwrap();

    //Create a file of random characters for the upload.
    let mut file = File::create(&key).expect("Could not create sample file.");
    // Loop until the file is 5 chunks.
    while file.metadata().unwrap().len() <= CHUNK_SIZE * 4 {
        let rand_string: String = thread_rng()
            .sample_iter(&Alphanumeric)
            .take(256)
            .map(char::from)
```

```
            .collect();
        let return_string: String = "\n".to_string();
        file.write_all(rand_string.as_ref())
            .expect("Error writing to file.");
        file.write_all(return_string.as_ref())
            .expect("Error writing to file.");
    }

    let path = Path::new(&key);
    let file_size = tokio::fs::metadata(path)
        .await
        .expect("it exists I swear")
        .len();

    let mut chunk_count = (file_size / CHUNK_SIZE) + 1;
    let mut size_of_last_chunk = file_size % CHUNK_SIZE;
    if size_of_last_chunk == 0 {
        size_of_last_chunk = CHUNK_SIZE;
        chunk_count -= 1;
    }

    if file_size == 0 {
        panic!("Bad file size.");
    }
    if chunk_count > MAX_CHUNKS {
        panic!("Too many chunks! Try increasing your chunk size.")
    }

    let mut upload_parts: Vec<CompletedPart> = Vec::new();

    for chunk_index in 0..chunk_count {
        let this_chunk = if chunk_count - 1 == chunk_index {
            size_of_last_chunk
        } else {
            CHUNK_SIZE
        };
        let stream = ByteStream::read_from()
            .path(path)
            .offset(chunk_index * CHUNK_SIZE)
            .length(Length::Exact(this_chunk))
            .build()
            .await
            .unwrap();
        //Chunk index needs to start at 0, but part numbers start at 1.
```

```
        let part_number = (chunk_index as i32) + 1;
        let upload_part_res = client
            .upload_part()
            .key(&key)
            .bucket(&bucket_name)
            .upload_id(upload_id)
            .body(stream)
            .part_number(part_number)
            .send()
            .await?;
        upload_parts.push(
            CompletedPart::builder()
                .e_tag(upload_part_res.e_tag.unwrap_or_default())
                .part_number(part_number)
                .build(),
        );
    }
    let completed_multipart_upload: CompletedMultipartUpload =
CompletedMultipartUpload::builder()
        .set_parts(Some(upload_parts))
        .build();

    let _complete_multipart_upload_res = client
        .complete_multipart_upload()
        .bucket(&bucket_name)
        .key(&key)
        .multipart_upload(completed_multipart_upload)
        .upload_id(upload_id)
        .send()
        .await
        .unwrap();

    let data: GetObjectOutput = s3_service::download_object(&client,
&bucket_name, &key).await?;
    let data_length: u64 = data
        .content_length()
        .unwrap_or_default()
        .try_into()
        .unwrap();
    if file.metadata().unwrap().len() == data_length {
        println!("Data lengths match.");
    } else {
        println!("The data was not the same size!");
    }
```

```
    s3_service::delete_objects(&client, &bucket_name)
        .await
        .expect("Error emptying bucket.");
    s3_service::delete_bucket(&client, &bucket_name)
        .await
        .expect("Error deleting bucket.");

    Ok(())
}
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Unggah aliran dengan ukuran yang tidak diketahui ke objek Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara mengunggah aliran dengan ukuran yang tidak diketahui ke objek Amazon S3.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Gunakan Klien S3 berbasis CRT AWS.

```
import com.example.s3.util.AsyncExampleUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.async.AsyncRequestBody;
import software.amazon.awssdk.core.async.BlockingInputStreamAsyncRequestBody;
import software.amazon.awssdk.core.exception.SdkException;
```

```
import software.amazon.awssdk.services.s3.S3AsyncClient;
import software.amazon.awssdk.services.s3.model.PutObjectResponse;

import java.io.ByteArrayInputStream;
import java.util.UUID;
import java.util.concurrent.CompletableFuture;

    /**
     * @param s33CrtAsyncClient - To upload content from a stream of unknown
 size, use the AWS CRT-based S3 client. For more information, see
     *                          https://docs.aws.amazon.com/sdk-for-java/latest/
developer-guide/crt-based-s3-client.html.
     * @param bucketName - The name of the bucket.
     * @param key - The name of the object.
     * @return software.amazon.awssdk.services.s3.model.PutObjectResponse -
 Returns metadata pertaining to the put object operation.
     */
    public PutObjectResponse putObjectFromStream(S3AsyncClient s33CrtAsyncClient,
 String bucketName, String key) {

        BlockingInputStreamAsyncRequestBody body =
                AsyncRequestBody.forBlockingInputStream(null); // 'null'
 indicates a stream will be provided later.

        CompletableFuture<PutObjectResponse> responseFuture =
                s33CrtAsyncClient.putObject(r -> r.bucket(bucketName).key(key),
 body);

        // AsyncExampleUtils.randomString() returns a random string up to 100
 characters.
        String randomString = AsyncExampleUtils.randomString();
        logger.info("random string to upload: {}: length={}", randomString,
 randomString.length());

        // Provide the stream of data to be uploaded.
        body.writeInputStream(new ByteArrayInputStream(randomString.getBytes()));

        PutObjectResponse response = responseFuture.join(); // Wait for the
 response.
        logger.info("Object {} uploaded to bucket {}.", key, bucketName);
        return response;
    }
}
```

Gunakan Manajer Transfer Amazon S3.

```
import com.example.s3.util.AsyncExampleUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.async.AsyncRequestBody;
import software.amazon.awssdk.core.async.BlockingInputStreamAsyncRequestBody;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.CompletedUpload;
import software.amazon.awssdk.transfer.s3.model.Upload;

import java.io.ByteArrayInputStream;
import java.util.UUID;

    /**
     * @param transferManager - To upload content from a stream of unknown size,
 use the S3TransferManager based on the AWS CRT-based S3 client.
     *                       For more information, see https://
docs.aws.amazon.com/sdk-for-java/latest/developer-guide/transfer-manager.html.
     * @param bucketName - The name of the bucket.
     * @param key - The name of the object.
     * @return - software.amazon.awssdk.transfer.s3.model.CompletedUpload - The
 result of the completed upload.
     */
    public CompletedUpload uploadStream(S3TransferManager transferManager, String
 bucketName, String key) {

        BlockingInputStreamAsyncRequestBody body =
                AsyncRequestBody.forBlockingInputStream(null); // 'null'
 indicates a stream will be provided later.

        Upload upload = transferManager.upload(builder -> builder
                .requestBody(body)
                .putObjectRequest(req -> req.bucket(bucketName).key(key))
                .build());

        // AsyncExampleUtils.randomString() returns a random string up to 100
 characters.
        String randomString = AsyncExampleUtils.randomString();
```

```
        logger.info("random string to upload: {}: length={}", randomString,
 randomString.length());

        // Provide the stream of data to be uploaded.
        body.writeInputStream(new ByteArrayInputStream(randomString.getBytes()));

        return upload.completionFuture().join();
    }
}
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Menggunakan checksum untuk bekerja dengan objek Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara menggunakan checksum untuk bekerja dengan objek Amazon S3.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Contoh kode menggunakan subset dari impor berikut.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.core.exception.SdkException;
import software.amazon.awssdk.core.sync.RequestBody;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.ChecksumAlgorithm;
```

```
import software.amazon.awssdk.services.s3.model.ChecksumMode;
import software.amazon.awssdk.services.s3.model.CompletedMultipartUpload;
import software.amazon.awssdk.services.s3.model.CompletedPart;
import software.amazon.awssdk.services.s3.model.CreateMultipartUploadResponse;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.s3.model.UploadPartRequest;
import software.amazon.awssdk.services.s3.model.UploadPartResponse;
import software.amazon.awssdk.services.s3.waiters.S3Waiter;
import software.amazon.awssdk.transfer.s3.S3TransferManager;
import software.amazon.awssdk.transfer.s3.model.FileUpload;
import software.amazon.awssdk.transfer.s3.model.UploadFileRequest;

import java.io.FileInputStream;
import java.io.IOException;
import java.io.RandomAccessFile;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.ByteBuffer;
import java.nio.file.Paths;
import java.security.DigestInputStream;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.Base64;
import java.util.List;
import java.util.Objects;
import java.util.UUID;
```

Tentukan algoritma checksum untuk metode `putObject` saat Anda membangun `PutObjectRequest`.

```
    public void putObjectWithChecksum() {
        s3Client.putObject(b -> b
                .bucket(bucketName)
                .key(key)
                .checksumAlgorithm(ChecksumAlgorithm.CRC32),
                RequestBody.fromString("This is a test"));
    }
```

Verifikasi checksum untuk `getObject` metode saat Anda membangun. GetObjectRequest

```
    public GetObjectResponse getObjectWithChecksum() {
        return s3Client.getObject(b -> b
                .bucket(bucketName)
                .key(key)
                .checksumMode(ChecksumMode.ENABLED))
                .response();
    }
```

Hitung terlebih dahulu checksum untuk metode `putObject` saat Anda membangun `PutObjectRequest`.

```
    public void putObjectWithPrecalculatedChecksum(String filePath) {
        String checksum = calculateChecksum(filePath, "SHA-256");

        s3Client.putObject((b -> b
                .bucket(bucketName)
                .key(key)
                .checksumSHA256(checksum)),
                RequestBody.fromFile(Paths.get(filePath)));
    }
```

Gunakan Manajer Transfer S3 di atas klien S3 AWS berbasis CRT untuk melakukan unggahan multibagian secara transparan ketika ukuran konten melebihi ambang batas. Ukuran ambang default adalah 8 MB.

Anda dapat menentukan algoritma checksum untuk SDK yang akan digunakan. Secara default, SDK menggunakan algoritma CRC32.

```
    public void multipartUploadWithChecksumTm(String filePath) {
        S3TransferManager transferManager = S3TransferManager.create();
        UploadFileRequest uploadFileRequest = UploadFileRequest.builder()
                .putObjectRequest(b -> b
                        .bucket(bucketName)
                        .key(key)
                        .checksumAlgorithm(ChecksumAlgorithm.SHA1))
                .source(Paths.get(filePath))
                .build();
        FileUpload fileUpload = transferManager.uploadFile(uploadFileRequest);
        fileUpload.completionFuture().join();
```

```
        transferManager.close();
    }
```

Gunakan S3Client API atau (S3 AsyncClient API) untuk melakukan upload multipart. Jika Anda menentukan checksum tambahan, Anda juga harus menentukan algoritma yang akan digunakan pada inisiasi unggahan. Anda juga harus menentukan algoritma untuk setiap permintaan bagian dan memberikan checksum yang dihitung untuk setiap bagian setelah diunggah.

```
    public void multipartUploadWithChecksumS3Client(String filePath) {
        ChecksumAlgorithm algorithm = ChecksumAlgorithm.CRC32;

        // Initiate the multipart upload.
        CreateMultipartUploadResponse createMultipartUploadResponse =
s3Client.createMultipartUpload(b -> b
                .bucket(bucketName)
                .key(key)
                .checksumAlgorithm(algorithm)); // Checksum specified on
initiation.
        String uploadId = createMultipartUploadResponse.uploadId();

        // Upload the parts of the file.
        int partNumber = 1;
        List<CompletedPart> completedParts = new ArrayList<>();
        ByteBuffer bb = ByteBuffer.allocate(1024 * 1024 * 5); // 5 MB byte buffer

        try (RandomAccessFile file = new RandomAccessFile(filePath, "r")) {
            long fileSize = file.length();
            long position = 0;
            while (position < fileSize) {
                file.seek(position);
                long read = file.getChannel().read(bb);

                bb.flip(); // Swap position and limit before reading from the
buffer.
                UploadPartRequest uploadPartRequest = UploadPartRequest.builder()
                        .bucket(bucketName)
                        .key(key)
                        .uploadId(uploadId)
                        .checksumAlgorithm(algorithm) // Checksum specified on
each part.
                        .partNumber(partNumber)
```

```
                        .build();

                UploadPartResponse partResponse = s3Client.uploadPart(
                        uploadPartRequest,
                        RequestBody.fromByteBuffer(bb));

                CompletedPart part = CompletedPart.builder()
                        .partNumber(partNumber)
                        .checksumCRC32(partResponse.checksumCRC32()) // Provide
the calculated checksum.
                        .eTag(partResponse.eTag())
                        .build();
                completedParts.add(part);

                bb.clear();
                position += read;
                partNumber++;
            }
        } catch (IOException e) {
            System.err.println(e.getMessage());
        }

        // Complete the multipart upload.
        s3Client.completeMultipartUpload(b -> b
                .bucket(bucketName)
                .key(key)
                .uploadId(uploadId)

.multipartUpload(CompletedMultipartUpload.builder().parts(completedParts).build()));
    }
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Java 2.x .
  - CompleteMultipartUpload
  - CreateMultipartUpload
  - UploadPart

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Bekerja dengan objek berversi Amazon S3 menggunakan SDK AWS

Contoh kode berikut ini menunjukkan cara:

- Membuat bucket S3 berversi.
- Mendapatkan semua versi objek.
- Mengembalikan objek ke versi sebelumnya.
- Menghapus dan memulihkan objek berversi.
- Menghapus semua versi objek secara permanen.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat fungsi yang membungkus tindakan S3.

```
def create_versioned_bucket(bucket_name, prefix):
    """
    Creates an Amazon S3 bucket, enables it for versioning, and configures a
 lifecycle
    that expires noncurrent object versions after 7 days.

    Adding a lifecycle configuration to a versioned bucket is a best practice.
    It helps prevent objects in the bucket from accumulating a large number of
    noncurrent versions, which can slow down request performance.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket_name: The name of the bucket to create.
    :param prefix: Identifies which objects are automatically expired under the
                   configured lifecycle rules.
    :return: The newly created bucket.
    """
```

```
    try:
        bucket = s3.create_bucket(
            Bucket=bucket_name,
            CreateBucketConfiguration={
                "LocationConstraint": s3.meta.client.meta.region_name
            },
        )
        logger.info("Created bucket %s.", bucket.name)
    except ClientError as error:
        if error.response["Error"]["Code"] == "BucketAlreadyOwnedByYou":
            logger.warning("Bucket %s already exists! Using it.", bucket_name)
            bucket = s3.Bucket(bucket_name)
        else:
            logger.exception("Couldn't create bucket %s.", bucket_name)
            raise

    try:
        bucket.Versioning().enable()
        logger.info("Enabled versioning on bucket %s.", bucket.name)
    except ClientError:
        logger.exception("Couldn't enable versioning on bucket %s.", bucket.name)
        raise

    try:
        expiration = 7
        bucket.LifecycleConfiguration().put(
            LifecycleConfiguration={
                "Rules": [
                    {
                        "Status": "Enabled",
                        "Prefix": prefix,
                        "NoncurrentVersionExpiration": {"NoncurrentDays":
 expiration},
                    }
                ]
            }
        )
        logger.info(
            "Configured lifecycle to expire noncurrent versions after %s days "
            "on bucket %s.",
            expiration,
            bucket.name,
        )
    except ClientError as error:
```

```
        logger.warning(
            "Couldn't configure lifecycle on bucket %s because %s. "
            "Continuing anyway.",
            bucket.name,
            error,
        )

    return bucket



def rollback_object(bucket, object_key, version_id):
    """
    Rolls back an object to an earlier version by deleting all versions that
    occurred after the specified rollback version.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that holds the object to roll back.
    :param object_key: The object to roll back.
    :param version_id: The version ID to roll back to.
    """
    # Versions must be sorted by last_modified date because delete markers are
    # at the end of the list even when they are interspersed in time.
    versions = sorted(
        bucket.object_versions.filter(Prefix=object_key),
        key=attrgetter("last_modified"),
        reverse=True,
    )

    logger.debug(
        "Got versions:\n%s",
        "\n".join(
            [
                f"\t{version.version_id}, last modified {version.last_modified}"
                for version in versions
            ]
        ),
    )

    if version_id in [ver.version_id for ver in versions]:
        print(f"Rolling back to version {version_id}")
        for version in versions:
```

```
            if version.version_id != version_id:
                version.delete()
                print(f"Deleted version {version.version_id}")
            else:
                break

        print(f"Active version is now {bucket.Object(object_key).version_id}")
    else:
        raise KeyError(
            f"{version_id} was not found in the list of versions for "
 f"{object_key}."
        )



def revive_object(bucket, object_key):
    """
    Revives a versioned object that was deleted by removing the object's active
    delete marker.
    A versioned object presents as deleted when its latest version is a delete
 marker.
    By removing the delete marker, we make the previous version the latest
 version
    and the object then presents as *not* deleted.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that contains the object.
    :param object_key: The object to revive.
    """
    # Get the latest version for the object.
    response = s3.meta.client.list_object_versions(
        Bucket=bucket.name, Prefix=object_key, MaxKeys=1
    )

    if "DeleteMarkers" in response:
        latest_version = response["DeleteMarkers"][0]
        if latest_version["IsLatest"]:
            logger.info(
                "Object %s was indeed deleted on %s. Let's revive it.",
                object_key,
                latest_version["LastModified"],
            )
```

```
            obj = bucket.Object(object_key)
            obj.Version(latest_version["VersionId"]).delete()
            logger.info(
                "Revived %s, active version is now %s  with body '%s'",
                object_key,
                obj.version_id,
                obj.get()["Body"].read(),
            )
        else:
            logger.warning(
                "Delete marker is not the latest version for %s!", object_key
            )
    elif "Versions" in response:
        logger.warning("Got an active version for %s, nothing to do.",
 object_key)
    else:
        logger.error("Couldn't get any version info for %s.", object_key)



def permanently_delete_object(bucket, object_key):
    """
    Permanently deletes a versioned object by deleting all of its versions.

    Usage is shown in the usage_demo_single_object function at the end of this
 module.

    :param bucket: The bucket that contains the object.
    :param object_key: The object to delete.
    """
    try:
        bucket.object_versions.filter(Prefix=object_key).delete()
        logger.info("Permanently deleted all versions of object %s.", object_key)
    except ClientError:
        logger.exception("Couldn't delete all versions of %s.", object_key)
        raise
```

Unggah bait puisi ke objek berversi dan lakukan serangkaian tindakan pada aaobjek tersebut.

```
def usage_demo_single_object(obj_prefix="demo-versioning/"):
```

```
    """
    Demonstrates usage of versioned object functions. This demo uploads a stanza
    of a poem and performs a series of revisions, deletions, and revivals on it.

    :param obj_prefix: The prefix to assign to objects created by this demo.
    """
    with open("father_william.txt") as file:
        stanzas = file.read().split("\n\n")

    width = get_terminal_size((80, 20))[0]
    print("-" * width)
    print("Welcome to the usage demonstration of Amazon S3 versioning.")
    print(
        "This demonstration uploads a single stanza of a poem to an Amazon "
        "S3 bucket and then applies various revisions to it."
    )
    print("-" * width)
    print("Creating a version-enabled bucket for the demo...")
    bucket = create_versioned_bucket("bucket-" + str(uuid.uuid1()), obj_prefix)

    print("\nThe initial version of our stanza:")
    print(stanzas[0])

    # Add the first stanza and revise it a few times.
    print("\nApplying some revisions to the stanza...")
    obj_stanza_1 = bucket.Object(f"{obj_prefix}stanza-1")
    obj_stanza_1.put(Body=bytes(stanzas[0], "utf-8"))
    obj_stanza_1.put(Body=bytes(stanzas[0].upper(), "utf-8"))
    obj_stanza_1.put(Body=bytes(stanzas[0].lower(), "utf-8"))
    obj_stanza_1.put(Body=bytes(stanzas[0][::-1], "utf-8"))
    print(
        "The latest version of the stanza is now:",
        obj_stanza_1.get()["Body"].read().decode("utf-8"),
        sep="\n",
    )

    # Versions are returned in order, most recent first.
    obj_stanza_1_versions =
bucket.object_versions.filter(Prefix=obj_stanza_1.key)
    print(
        "The version data of the stanza revisions:",
        *[
            f"    {version.version_id}, last modified {version.last_modified}"
            for version in obj_stanza_1_versions
```

```
        ],
        sep="\n",
    )

    # Rollback two versions.
    print("\nRolling back two versions...")
    rollback_object(bucket, obj_stanza_1.key, list(obj_stanza_1_versions)
[2].version_id)
    print(
        "The latest version of the stanza:",
        obj_stanza_1.get()["Body"].read().decode("utf-8"),
        sep="\n",
    )

    # Delete the stanza
    print("\nDeleting the stanza...")
    obj_stanza_1.delete()
    try:
        obj_stanza_1.get()
    except ClientError as error:
        if error.response["Error"]["Code"] == "NoSuchKey":
            print("The stanza is now deleted (as expected).")
        else:
            raise

    # Revive the stanza
    print("\nRestoring the stanza...")
    revive_object(bucket, obj_stanza_1.key)
    print(
        "The stanza is restored! The latest version is again:",
        obj_stanza_1.get()["Body"].read().decode("utf-8"),
        sep="\n",
    )

    # Permanently delete all versions of the object. This cannot be undone!
    print("\nPermanently deleting all versions of the stanza...")
    permanently_delete_object(bucket, obj_stanza_1.key)
    obj_stanza_1_versions =
 bucket.object_versions.filter(Prefix=obj_stanza_1.key)
    if len(list(obj_stanza_1_versions)) == 0:
        print("The stanza has been permanently deleted and now has no versions.")
    else:
        print("Something went wrong. The stanza still exists!")
```

```
    print(f"\nRemoving {bucket.name}...")
    bucket.delete()
    print(f"{bucket.name} deleted.")
    print("Demo done!")
```

- Untuk detail API, lihat topik berikut ini adalah Referensi API SDK untuk Python (Boto3)AWS .
  - CreateBucket
  - DeleteObject
  - ListObjectVersions
  - PutBucketLifecycleConfiguration

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Contoh tanpa server untuk Amazon S3 menggunakan SDK AWS

Contoh kode berikut menunjukkan cara menggunakan Amazon S3 dengan AWS SDK.

Contoh

- Menginvokasi fungsi Lambda dari pemicu Amazon S3

Menginvokasi fungsi Lambda dari pemicu Amazon S3

Contoh kode berikut menunjukkan cara mengimplementasikan fungsi Lambda yang menerima peristiwa yang dipicu dengan mengunggah objek ke bucket S3. Fungsi ini mengambil nama bucket S3 dan kunci objek dari parameter peristiwa dan memanggil Amazon S3 API untuk mengambil dan mencatat jenis konten objek.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori contoh Nirserver.

Menggunakan peristiwa S3 dengan Lambda menggunakan.NET.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
using System.Threading.Tasks;
using Amazon.Lambda.Core;
using Amazon.S3;
using System;
using Amazon.Lambda.S3Events;
using System.Web;

// Assembly attribute to enable the Lambda function's JSON input to be converted
 into a .NET class.
[assembly:
 LambdaSerializer(typeof(Amazon.Lambda.Serialization.SystemTextJson.DefaultLambdaJsonSeri

namespace S3Integration
{
    public class Function
    {
        private static AmazonS3Client _s3Client;
        public Function() : this(null)
        {
        }

        internal Function(AmazonS3Client s3Client)
        {
            _s3Client = s3Client ?? new AmazonS3Client();
        }

        public async Task<string> Handler(S3Event evt, ILambdaContext context)
        {
            try
```

```
            {
                if (evt.Records.Count <= 0)
                {
                    context.Logger.LogLine("Empty S3 Event received");
                    return string.Empty;
                }

                var bucket = evt.Records[0].S3.Bucket.Name;
                var key = HttpUtility.UrlDecode(evt.Records[0].S3.Object.Key);

                context.Logger.LogLine($"Request is for {bucket} and {key}");

                var objectResult = await _s3Client.GetObjectAsync(bucket, key);

                context.Logger.LogLine($"Returning {objectResult.Key}");

                return objectResult.Key;
            }
            catch (Exception e)
            {
                context.Logger.LogLine($"Error processing request -
 {e.Message}");

                return string.Empty;
            }
        }
    }
}
```

Go

SDK untuk Go V2

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori contoh Nirserver.

Menggunakan peristiwa S3 dengan Lambda menggunakan Go.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
```

```
// SPDX-License-Identifier: Apache-2.0
package main

import (
 "context"
 "log"

 "github.com/aws/aws-lambda-go/events"
 "github.com/aws/aws-lambda-go/lambda"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/s3"
)

func handler(ctx context.Context, s3Event events.S3Event) error {
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
  log.Printf("failed to load default config: %s", err)
  return err
 }
 s3Client := s3.NewFromConfig(sdkConfig)

 for _, record := range s3Event.Records {
  bucket := record.S3.Bucket.Name
  key := record.S3.Object.URLDecodedKey
  headOutput, err := s3Client.HeadObject(ctx, &s3.HeadObjectInput{
   Bucket: &bucket,
   Key:    &key,
  })
  if err != nil {
   log.Printf("error getting head of object %s/%s: %s", bucket, key, err)
   return err
  }
  log.Printf("successfully retrieved %s/%s of type %s", bucket, key,
 *headOutput.ContentType)
 }

 return nil
}

func main() {
 lambda.Start(handler)
}
```

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori contoh Nirserver.

Menggunakan peristiwa S3 dengan Lambda menggunakan Java.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
package example;

import software.amazon.awssdk.services.s3.model.HeadObjectRequest;
import software.amazon.awssdk.services.s3.model.HeadObjectResponse;
import software.amazon.awssdk.services.s3.S3Client;

import com.amazonaws.services.lambda.runtime.Context;
import com.amazonaws.services.lambda.runtime.RequestHandler;
import com.amazonaws.services.lambda.runtime.events.S3Event;
import
 com.amazonaws.services.lambda.runtime.events.models.s3.S3EventNotification.S3EventNotifi

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class Handler implements RequestHandler<S3Event, String> {
    private static final Logger logger = LoggerFactory.getLogger(Handler.class);
    @Override
    public String handleRequest(S3Event s3event, Context context) {
        try {
          S3EventNotificationRecord record = s3event.getRecords().get(0);
          String srcBucket = record.getS3().getBucket().getName();
          String srcKey = record.getS3().getObject().getUrlDecodedKey();

          S3Client s3Client = S3Client.builder().build();
          HeadObjectResponse headObject = getHeadObject(s3Client, srcBucket,
 srcKey);
```

```
            logger.info("Successfully retrieved " + srcBucket + "/" + srcKey + " of
 type " + headObject.contentType());

            return "Ok";
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }

    private HeadObjectResponse getHeadObject(S3Client s3Client, String bucket,
 String key) {
        HeadObjectRequest headObjectRequest = HeadObjectRequest.builder()
                .bucket(bucket)
                .key(key)
                .build();
        return s3Client.headObject(headObjectRequest);
    }
}
```

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori contoh Nirserver.

Mengkonsumsi acara S3 dengan menggunakan JavaScript Lambda.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { S3Client, HeadObjectCommand } from "@aws-sdk/client-s3";

const client = new S3Client();

exports.handler = async (event, context) => {

    // Get the object from the event and show its content type
    const bucket = event.Records[0].s3.bucket.name;
```

```
    const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g,
 ' '));

    try {
        const { ContentType } = await client.send(new HeadObjectCommand({
            Bucket: bucket,
            Key: key,
        }));

        console.log('CONTENT TYPE:', ContentType);
        return ContentType;

    } catch (err) {
        console.log(err);
        const message = `Error getting object ${key} from bucket ${bucket}. Make
 sure they exist and your bucket is in the same region as this function.`;
        console.log(message);
        throw new Error(message);
    }
};
```

Mengkonsumsi acara S3 dengan menggunakan TypeScript Lambda.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { S3Event } from 'aws-lambda';
import { S3Client, HeadObjectCommand } from '@aws-sdk/client-s3';

const s3 = new S3Client({ region: process.env.AWS_REGION });

export const handler = async (event: S3Event): Promise<string | undefined> => {
  // Get the object from the event and show its content type
  const bucket = event.Records[0].s3.bucket.name;
  const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, '
 '));
  const params = {
    Bucket: bucket,
    Key: key,
  };
  try {
    const { ContentType } = await s3.send(new HeadObjectCommand(params));
    console.log('CONTENT TYPE:', ContentType);
```

```
    return ContentType;
  } catch (err) {
    console.log(err);
    const message = `Error getting object ${key} from bucket ${bucket}. Make sure
 they exist and your bucket is in the same region as this function.`;
    console.log(message);
    throw new Error(message);
  }
};
```

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori contoh Nirserver.

Mengkonsumsi acara S3 dengan Lambda menggunakan PHP.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
<?php

use Bref\Context\Context;
use Bref\Event\S3\S3Event;
use Bref\Event\S3\S3Handler;
use Bref\Logger\StderrLogger;

require __DIR__ . '/vendor/autoload.php';


class Handler extends S3Handler
{
    private StderrLogger $logger;
    public function __construct(StderrLogger $logger)
    {
        $this->logger = $logger;
    }
```

```
    public function handleS3(S3Event $event, Context $context) : void
    {
        $this->logger->info("Processing S3 records");

        // Get the object from the event and show its content type
        $records = $event->getRecords();

        foreach ($records as $record)
        {
            $bucket = $record->getBucket()->getName();
            $key = urldecode($record->getObject()->getKey());

            try {
                $fileSize = urldecode($record->getObject()->getSize());
                echo "File Size: " . $fileSize . "\n";
                // TODO: Implement your custom processing logic here
            } catch (Exception $e) {
                echo $e->getMessage() . "\n";
                echo 'Error getting object ' . $key . ' from bucket ' .
 $bucket . '. Make sure they exist and your bucket is in the same region as this
 function.' . "\n";
                throw $e;
            }
        }
    }
}

$logger = new StderrLogger();
return new Handler($logger);
```

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori contoh Nirserver.

Menggunakan peristiwa S3 dengan Lambda menggunakan Python.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
import json
import urllib.parse
import boto3

print('Loading function')

s3 = boto3.client('s3')


def lambda_handler(event, context):
    #print("Received event: " + json.dumps(event, indent=2))

    # Get the object from the event and show its content type
    bucket = event['Records'][0]['s3']['bucket']['name']
    key = urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'],
 encoding='utf-8')
    try:
        response = s3.get_object(Bucket=bucket, Key=key)
        print("CONTENT TYPE: " + response['ContentType'])
        return response['ContentType']
    except Exception as e:
        print(e)
        print('Error getting object {} from bucket {}. Make sure they exist and
 your bucket is in the same region as this function.'.format(key, bucket))
        raise e
```

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori contoh Nirserver.

Mengkonsumsi acara S3 dengan Lambda menggunakan Ruby.

```
require 'json'
require 'uri'
require 'aws-sdk'

puts 'Loading function'

def lambda_handler(event:, context:)
  s3 = Aws::S3::Client.new(region: 'region') # Your AWS region
  # puts "Received event: #{JSON.dump(event)}"

  # Get the object from the event and show its content type
  bucket = event['Records'][0]['s3']['bucket']['name']
  key = URI.decode_www_form_component(event['Records'][0]['s3']['object']['key'],
 Encoding::UTF_8)
  begin
    response = s3.get_object(bucket: bucket, key: key)
    puts "CONTENT TYPE: #{response.content_type}"
    return response.content_type
  rescue StandardError => e
    puts e.message
    puts "Error getting object #{key} from bucket #{bucket}. Make sure they exist
 and your bucket is in the same region as this function."
    raise e
  end
end
```

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori contoh Nirserver.

Menggunakan peristiwa S3 dengan Lambda menggunakan Rust.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
```

```
use aws_lambda_events::event::s3::S3Event;
use aws_sdk_s3::{Client};
use lambda_runtime::{run, service_fn, Error, LambdaEvent};


/// Main function
#[tokio::main]
async fn main() -> Result<(), Error> {
    tracing_subscriber::fmt()
        .with_max_level(tracing::Level::INFO)
        .with_target(false)
        .without_time()
        .init();

    // Initialize the AWS SDK for Rust
    let config = aws_config::load_from_env().await;
    let s3_client = Client::new(&config);

    let res = run(service_fn(|request: LambdaEvent<S3Event>| {
        function_handler(&s3_client, request)
    })).await;

    res
}

async fn function_handler(
    s3_client: &Client,
    evt: LambdaEvent<S3Event>
) -> Result<(), Error> {
    tracing::info!(records = ?evt.payload.records.len(), "Received request from
 SQS");

    if evt.payload.records.len() == 0 {
        tracing::info!("Empty S3 event received");
    }

    let bucket = evt.payload.records[0].s3.bucket.name.as_ref().expect("Bucket
 name to exist");
    let key = evt.payload.records[0].s3.object.key.as_ref().expect("Object key to
 exist");

    tracing::info!("Request is for {} and object {}", bucket, key);

    let s3_get_object_result = s3_client
```

```
        .get_object()
        .bucket(bucket)
        .key(key)
        .send()
        .await;

    match s3_get_object_result {
        Ok(_) => tracing::info!("S3 Get Object success, the s3GetObjectResult
 contains a 'body' property of type ByteStream"),
        Err(_) => tracing::info!("Failure with S3 Get Object request")
    }

    Ok(())
}
```

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Contoh lintas layanan untuk Amazon AWS S3 menggunakan SDK

Contoh aplikasi berikut menggunakan AWS SDK untuk menggabungkan Amazon S3 dengan yang lain. Layanan AWS Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan petunjuk tentang cara mengatur dan menjalankan aplikasi.

Contoh

- Membangun aplikasi Amazon Transcribe
- Mengonversi teks menjadi ucapan dan kembali ke teks menggunakan AWS SDK
- Membuat aplikasi manajemen aset foto yang memungkinkan pengguna mengelola foto menggunakan label
- Membuat aplikasi penjelajah Amazon Textract
- Mendeteksi APD dalam gambar dengan Amazon AWS Rekognition menggunakan SDK
- Mendeteksi entitas dalam teks yang diekstrak dari gambar menggunakan SDK AWS
- Mendeteksi wajah dalam gambar menggunakan AWS SDK
- Mendeteksi objek dalam gambar dengan Amazon Rekognition menggunakan SDK AWS
- Mendeteksi orang dan objek dalam video dengan Amazon Rekognition menggunakan SDK AWS

- Simpan EXIF dan informasi gambar lainnya menggunakan SDK AWS
- Transformasi data untuk aplikasi Anda dengan S3 Object Lambda

Membangun aplikasi Amazon Transcribe

Contoh kode berikut ini menunjukkan cara menggunakan Amazon Transcribe untuk menyalin dan menampilkan rekaman suara di peramban.

JavaScript

SDK untuk JavaScript (v3)

Buat aplikasi yang menggunakan Amazon Transcribe untuk menyalin dan menampilkan rekaman suara di peramban. Aplikasi ini menggunakan dua bucket Amazon Simple Storage Service (Amazon S3), satu untuk meng-host kode aplikasi, dan satu lagi untuk menyimpan transkripsi. Aplikasi ini menggunakan kolam pengguna Amazon Cognito untuk mengautentikasi pengguna Anda. Pengguna yang diautentikasi memiliki izin AWS Identity and Access Management (IAM) untuk mengakses layanan yang diperlukan. AWS

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Contoh ini juga tersedia di panduan developer v3 AWS SDK for JavaScript.

Layanan yang digunakan dalam contoh ini

- Identitas Amazon Cognito
- Amazon S3
- Amazon Transcribe

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mengonversi teks menjadi ucapan dan kembali ke teks menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara:

- Menggunakan Amazon Polly untuk mensintesis file input teks biasa (UTF-8) ke file audio.
- Mengunggah file audio ke bucket Amazon S3.
- Menggunakan Amazon Transcribe untuk mengonversi file audio menjadi teks.
- Tampilkan teks.

Rust

SDK untuk Rust

Gunakan Amazon Polly untuk mensintesis file input teks biasa (UTF-8) menjadi file audio, unggah file audio ke bucket Amazon S3, gunakan Amazon Transcribe untuk mengonversi file audio tersebut menjadi teks, dan tampilkan teksnya.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Amazon Polly
- Amazon S3
- Amazon Transcribe

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Membuat aplikasi manajemen aset foto yang memungkinkan pengguna mengelola foto menggunakan label

Contoh kode berikut ini menunjukkan cara membuat aplikasi nirserver yang memungkinkan pengguna mengelola foto menggunakan label.

.NET

AWS SDK for .NET

Menunjukkan cara mengembangkan aplikasi manajemen aset foto yang mendeteksi label dalam gambar menggunakan Amazon Rekognition dan menyimpannya untuk pengambilan nanti.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Untuk mendalami tentang asal usul contoh ini, lihat postingan di Komunitas AWS.

Layanan yang digunakan dalam contoh ini

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

C++

SDK untuk C++

Menunjukkan cara mengembangkan aplikasi manajemen aset foto yang mendeteksi label dalam gambar menggunakan Amazon Rekognition dan menyimpannya untuk pengambilan nanti.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Untuk mendalami tentang asal usul contoh ini, lihat postingan di Komunitas AWS.

Layanan yang digunakan dalam contoh ini

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

Java

SDK untuk Java 2.x

Menunjukkan cara mengembangkan aplikasi manajemen aset foto yang mendeteksi label dalam gambar menggunakan Amazon Rekognition dan menyimpannya untuk pengambilan nanti.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Untuk mendalami tentang asal usul contoh ini, lihat postingan di Komunitas AWS.

Layanan yang digunakan dalam contoh ini

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

JavaScript

SDK untuk JavaScript (v3)

Menunjukkan cara mengembangkan aplikasi manajemen aset foto yang mendeteksi label dalam gambar menggunakan Amazon Rekognition dan menyimpannya untuk pengambilan nanti.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Untuk mendalami tentang asal usul contoh ini, lihat postingan di Komunitas AWS.

Layanan yang digunakan dalam contoh ini

- API Gateway
- DynamoDB

- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

Kotlin

SDK untuk Kotlin

Menunjukkan cara mengembangkan aplikasi manajemen aset foto yang mendeteksi label dalam gambar menggunakan Amazon Rekognition dan menyimpannya untuk pengambilan nanti.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Untuk mendalami tentang asal usul contoh ini, lihat postingan di Komunitas AWS.

Layanan yang digunakan dalam contoh ini

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

PHP

SDK untuk PHP

Menunjukkan cara mengembangkan aplikasi manajemen aset foto yang mendeteksi label dalam gambar menggunakan Amazon Rekognition dan menyimpannya untuk pengambilan nanti.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Untuk mendalami tentang asal usul contoh ini, lihat postingan di Komunitas AWS.

Layanan yang digunakan dalam contoh ini

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

Rust

SDK untuk Rust

Menunjukkan cara mengembangkan aplikasi manajemen aset foto yang mendeteksi label dalam gambar menggunakan Amazon Rekognition dan menyimpannya untuk pengambilan nanti.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Untuk mendalami tentang asal usul contoh ini, lihat postingan di Komunitas AWS.

Layanan yang digunakan dalam contoh ini

- API Gateway
- DynamoDB
- Lambda
- Amazon Rekognition
- Amazon S3
- Amazon SNS

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Membuat aplikasi penjelajah Amazon Textract

Contoh kode berikut ini menunjukkan cara menjelajahi output Amazon Textract melalui aplikasi interaktif.

JavaScript

SDK untuk JavaScript (v3)

Menunjukkan cara menggunakan aplikasi AWS SDK for JavaScript untuk membangun aplikasi React yang menggunakan Amazon Textract untuk mengekstrak data dari gambar dokumen dan menampilkannya di halaman web interaktif. Contoh ini berjalan di peramban web dan memerlukan identitas Amazon Cognito yang diautentikasi sebagai kredensialnya. Contoh ini menggunakan Amazon Simple Storage Service (Amazon S3) untuk penyimpanan, dan untuk notifikasi, contoh ini mengambil polling antrean Amazon Simple Queue Service (Amazon SQS) yang berlangganan topik Amazon Simple Notification Service (Amazon SNS).

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Identitas Amazon Cognito
- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Python

SDK untuk Python (Boto3)

Menunjukkan cara menggunakan Amazon Textract untuk mendeteksi elemen teks, formulir, dan tabel dalam gambar dokumen. AWS SDK for Python (Boto3) Gambar input dan output Amazon Textract ditampilkan dalam aplikasi Tkinter yang memungkinkan Anda menjelajahi elemen yang terdeteksi.

- Kirim gambar dokumen ke Amazon Textract dan jelajahi output elemen yang terdeteksi.
- Kirim gambar langsung ke Amazon Textract atau melalui bucket Amazon Simple Storage Service (Amazon S3).

- Gunakan API asinkron untuk memulai tugas yang menerbitkan notifikasi ke topik Amazon Simple Notification Service (Amazon SNS) saat tugas selesai.
- Lakukan polling pada antrean Amazon Simple Queue Service (Amazon SQS) untuk mendapatkan pesan penyelesaian tugas dan tampilkan hasilnya.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mendeteksi APD dalam gambar dengan Amazon AWS Rekognition menggunakan SDK

Contoh kode berikut menunjukkan cara membuat aplikasi yang menggunakan Amazon Rekognition untuk mendeteksi Alat Pelindung Diri (APD) dalam gambar.

Java

SDK untuk Java 2.x

Menunjukkan cara membuat AWS Lambda fungsi yang mendeteksi gambar dengan Alat Pelindung Diri.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- DynamoDB

- Amazon Rekognition
- Amazon S3
- Amazon SES

JavaScript

SDK untuk JavaScript (v3)

Menunjukkan cara menggunakan Amazon Rekognition dengan AWS SDK for JavaScript membuat aplikasi untuk mendeteksi alat pelindung diri (APD) pada gambar yang terletak di bucket Amazon Simple Storage Service (Amazon S3). Aplikasi tersebut menyimpan hasilnya ke tabel Amazon DynamoDB, dan mengirimkan notifikasi email kepada admin beserta hasilnya menggunakan Amazon Simple Email Service (Amazon SES).

Pelajari cara:

- Membuat pengguna yang tidak diautentikasi menggunakan Amazon Cognito.
- Menganalisis gambar untuk APD menggunakan Amazon Rekognition.
- Memverifikasi alamat email untuk Amazon SES.
- Memperbarui tabel DynamoDB dengan hasil.
- Mengirim notifikasi email menggunakan Amazon SES.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- DynamoDB
- Amazon Rekognition
- Amazon S3
- Amazon SES

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mendeteksi entitas dalam teks yang diekstrak dari gambar menggunakan SDK AWS

Contoh kode berikut menunjukkan cara menggunakan Amazon Comprehend untuk mendeteksi entitas dalam teks yang diekstrak oleh Amazon Textract dari gambar yang disimpan di Amazon S3.

Python

SDK untuk Python (Boto3)

Menunjukkan cara menggunakan AWS SDK for Python (Boto3) dalam buku catatan Jupyter untuk mendeteksi entitas dalam teks yang diekstraksi dari gambar. Contoh ini menggunakan Amazon Textract untuk mengekstrak teks dari gambar yang disimpan di Amazon Simple Storage Service (Amazon S3) dan Amazon Comprehend untuk mendeteksi entitas dalam teks yang diekstraksi.

Contoh ini adalah notebook Jupyter dan harus dijalankan di lingkungan yang dapat meng-host notebook. Untuk petunjuk tentang cara menjalankan contoh menggunakan Amazon SageMaker, lihat petunjuk di TextractAndComprehendNotebook.ipynb.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Amazon Comprehend
- Amazon S3
- Amazon Textract

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mendeteksi wajah dalam gambar menggunakan AWS SDK

Contoh kode berikut ini menunjukkan cara:

- Menyimpan gambar di bucket Amazon S3.

- Menggunakan Amazon Rekognition untuk mendeteksi detail wajah, seperti rentang usia, jenis kelamin, dan emosi (seperti tersenyum).
- Menampilkan detail tersebut.

Rust

SDK untuk Rust

Menyimpan gambar di bucket Amazon S3 dengan prefiks unggahan, menggunakan Amazon Rekognition untuk mendeteksi detail wajah, seperti rentang usia, jenis kelamin, dan emosi (tersenyum, dll.), dan menampilkan detail tersebut.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Amazon Rekognition
- Amazon S3

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mendeteksi objek dalam gambar dengan Amazon Rekognition menggunakan SDK AWS

Contoh kode berikut menunjukkan cara membuat aplikasi yang menggunakan Amazon Rekognition untuk mendeteksi objek berdasarkan kategori dalam gambar.

.NET

AWS SDK for .NET

Menunjukkan cara menggunakan Amazon Rekognition .NET API untuk membuat aplikasi yang menggunakan Amazon Rekognition untuk mengidentifikasi objek berdasarkan kategori dalam gambar yang berada di bucket Amazon Simple Storage Service (Amazon S3). Aplikasi ini mengirimkan notifikasi email kepada admin beserta hasilnya menggunakan Amazon Simple Email Service (Amazon SES).

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Amazon Rekognition
- Amazon S3
- Amazon SES

Java

SDK untuk Java 2.x

Menunjukkan cara menggunakan Amazon Rekognition Java API untuk membuat aplikasi yang menggunakan Amazon Rekognition untuk mengidentifikasi objek berdasarkan kategori dalam gambar yang terletak di bucket Amazon Simple Storage Service (Amazon S3). Aplikasi ini mengirimkan notifikasi email kepada admin beserta hasilnya menggunakan Amazon Simple Email Service (Amazon SES).

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Amazon Rekognition
- Amazon S3
- Amazon SES

JavaScript

SDK untuk JavaScript (v3)

Menunjukkan cara menggunakan Amazon Rekognition dengan membuat aplikasi AWS SDK for JavaScript yang menggunakan Amazon Rekognition untuk mengidentifikasi objek berdasarkan kategori dalam gambar yang terletak di bucket Amazon Simple Storage Service (Amazon S3). Aplikasi ini mengirimkan notifikasi email kepada admin beserta hasilnya menggunakan Amazon Simple Email Service (Amazon SES).

Pelajari cara:

- Membuat pengguna yang tidak diautentikasi menggunakan Amazon Cognito.
- Menganalisi gambar untuk objek menggunakan Amazon Rekognition.
- Memverifikasi alamat email untuk Amazon SES.
- Mengirim notifikasi email menggunakan Amazon SES.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Amazon Rekognition
- Amazon S3
- Amazon SES

Kotlin

SDK untuk Kotlin

Menunjukkan cara menggunakan Amazon Rekognition Kotlin API untuk membuat aplikasi yang menggunakan Amazon Rekognition untuk mengidentifikasi objek berdasarkan kategori dalam gambar yang berada di bucket Amazon Simple Storage Service (Amazon S3). Aplikasi ini mengirimkan notifikasi email kepada admin beserta hasilnya menggunakan Amazon Simple Email Service (Amazon SES).

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Amazon Rekognition
- Amazon S3
- Amazon SES

Python

SDK untuk Python (Boto3)

Menunjukkan cara menggunakan AWS SDK for Python (Boto3) untuk membuat aplikasi web yang memungkinkan Anda melakukan hal berikut:

- Mengunggah foto ke bucket Amazon Simple Storage Service (Amazon S3).
- Menggunakan Amazon Rekognition untuk menganalisis dan memberi label pada foto.
- Menggunakan Amazon Simple Email Service (Amazon SES) untuk mengirim laporan email analisis gambar.

Contoh ini berisi dua komponen utama: halaman web yang ditulis di dalamnya JavaScript yang dibangun dengan React, dan layanan REST yang ditulis dengan Python yang dibangun dengan Flask-RESTful.

Anda dapat menggunakan halaman web React untuk:

- Menampilkan daftar gambar yang disimpan di bucket S3 Anda.
- Mengunggah gambar dari komputer ke bucket S3.
- Menampilkan gambar dan label yang mengidentifikasi item yang terdeteksi dalam gambar.
- Mendapatkan laporan semua gambar di bucket S3 Anda dan mengirimkan email laporan tersebut.

Halaman web memanggil layanan REST. Layanan mengirimkan permintaan ke AWS untuk melakukan tindakan berikut:

- Mendapatkan dan memfilter daftar gambar dalam bucket S3 Anda.
- Merngunggah foto ke bucket S3 Anda.
- Menggunakan Amazon Rekognition untuk menganalisis foto individual dan mendapatkan daftar label yang mengidentifikasi item yang terdeteksi dalam foto.
- Menganlisis semua foto di bucket S3 Anda dan menggunakan Amazon SES untuk mengirim laporan melalui email.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Amazon Rekognition
- Amazon S3
- Amazon SES

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Mendeteksi orang dan objek dalam video dengan Amazon Rekognition menggunakan SDK AWS

Contoh kode berikut menunjukkan cara mendeteksi orang dan objek dalam video dengan Amazon Rekognition.

Java

SDK untuk Java 2.x

Menunjukkan cara menggunakan Amazon Rekognition Java API untuk membuat aplikasi guna mendeteksi wajah dan objek di video yang berada di bucket Amazon Simple Storage Service (Amazon S3). Aplikasi ini mengirimkan notifikasi email kepada admin beserta hasilnya menggunakan Amazon Simple Email Service (Amazon SES).

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Amazon Rekognition
- Amazon S3
- Amazon SES

JavaScript

SDK untuk JavaScript (v3)

Menunjukkan cara menggunakan Amazon Rekognition dengan AWS SDK for JavaScript membuat aplikasi untuk mendeteksi wajah dan objek dalam video yang terletak di bucket Amazon Simple Storage Service (Amazon S3). Aplikasi ini mengirimkan notifikasi email kepada admin beserta hasilnya menggunakan Amazon Simple Email Service (Amazon SES).

Pelajari cara:

- Membuat pengguna yang tidak diautentikasi menggunakan Amazon Cognito.
- Menganalisis gambar untuk APD menggunakan Amazon Rekognition.
- Memverifikasi alamat email untuk Amazon SES.
- Mengirim notifikasi email menggunakan Amazon SES.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Amazon Rekognition
- Amazon S3
- Amazon SES

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Simpan EXIF dan informasi gambar lainnya menggunakan SDK AWS

Contoh kode berikut ini menunjukkan cara:

- Mendapatkan informasi EXIF dari file JPG, JPEG, atau PNG.
- Mengunggah file gambar ke bucket Amazon S3.
- Menggunakan Amazon Rekognition untuk mengidentifikasi tiga atribut teratas (label) dalam file.
- Menambahkan informasi EXIF dan label ke tabel Amazon DynamoDB di Wilayah.

Rust

SDK untuk Rust

Mendapatkan informasi EXIF dari file JPG, JPEG, atau PNG, mengunggah file gambar ke bucket Amazon S3, menggunakan Amazon Rekognition untuk mengidentifikasi tiga atribut teratas (label di Amazon Rekognition) dalam file, dan menambahkan EXIF dan informasi label ke tabel Amazon DynamoDB di Wilayah.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- DynamoDB
- Amazon Rekognition

- Amazon S3

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Transformasi data untuk aplikasi Anda dengan S3 Object Lambda

Contoh kode berikut menunjukkan cara mengubah data untuk aplikasi Anda dengan S3 Object Lambda.

.NET

AWS SDK for .NET

Menunjukkan cara menambahkan kode kustom ke permintaan GET S3 standar untuk memodifikasi objek yang diminta diambil dari S3 sehingga objek sesuai dengan kebutuhan klien atau aplikasi yang meminta.

Untuk kode sumber lengkap dan instruksi tentang cara mengatur dan menjalankan, lihat contoh lengkapnya di GitHub.

Layanan yang digunakan dalam contoh ini

- Lambda
- Amazon S3

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Pemecahan Masalah

Bagian ini menjelaskan cara mengatasi masalah Amazon S3 dan menjelaskan cara mendapatkan ID permintaan yang akan Anda butuhkan saat Anda menghubungi AWS Support.

Topik

- Pecahkan masalah kesalahan Akses Ditolak (403 Forbidden) di Amazon S3
- Memecahkan Masalah Operasi Batch
- Penyelesaian masalah CORS
- Memecahkan masalah Siklus Hidup S3
- Memecahkan masalah replikasi
- Memecahkan masalah pencatatan akses server
- Pecahkan masalah Penentuan Versi
- Mendapatkan ID permintaan Amazon S3 untuk AWS Support

Pecahkan masalah kesalahan Akses Ditolak (403 Forbidden) di Amazon S3

> **Important**
>
> Pada 13 Mei 2024, kami mulai menerapkan perubahan untuk menghilangkan biaya atas permintaan tidak sah yang tidak diprakarsai oleh pemilik bucket. Setelah penerapan perubahan ini selesai, pemilik bucket tidak akan pernah dikenakan biaya permintaan atau bandwidth untuk permintaan yang mengembalikan kesalahan `AccessDenied` (HTTP`403 Forbidden`) saat permintaan ini dimulai dari luar AWS akun atau organisasi masing-masing. AWS Untuk informasi selengkapnya tentang daftar lengkap HTTP `3XX` dan kode `4XX` status yang tidak akan ditagih, lihatPenagihan untuk respons kesalahan Amazon S3. Perubahan penagihan ini tidak memerlukan pembaruan untuk aplikasi Anda dan berlaku untuk semua bucket S3. Ketika penerapan perubahan ini selesai di semua Wilayah AWS, kami akan memperbarui dokumentasi kami.

Topik berikut mencakup penyebab paling umum kesalahan Akses Ditolak (403 Forbidden) di Amazon S3.

> **Note**
>
> Untuk `Access Denied` (HTTP`403 Forbidden`), S3 tidak membebankan biaya kepada pemilik bucket saat permintaan dimulai di luar AWS akun individu pemilik bucket atau organisasi pemilik bucket. AWS

Topik

- Kebijakan bucket dan kebijakan IAM
- Pengaturan Amazon S3 ACL
- Pengaturan S3 Block Public Access
- Pengaturan enkripsi Amazon S3
- Pengaturan Kunci Objek S3
- Kebijakan titik akhir VPC
- AWS Organizations kebijakan
- Pengaturan titik akses

> **Note**
>
> Jika Anda mencoba memecahkan masalah izin, mulailah dengan bagian Kebijakan bucket dan kebijakan IAM, dan pastikan untuk mengikuti panduan di Tips untuk memeriksa izin.

Kebijakan bucket dan kebijakan IAM

Operasi tingkat bucket

Jika tidak ada kebijakan bucket, maka bucket secara implisit mengizinkan permintaan dari identitas AWS Identity and Access Management (IAM) apa pun di akun pemilik ember. Bucket juga secara implisit menolak permintaan dari identitas IAM lainnya dari akun lain, serta menolak permintaan anonim (tidak ditandatangani). Namun, jika tidak ada kebijakan pengguna IAM, peminta (kecuali mereka adalah pengguna root) secara implisit ditolak dalam membuat permintaan apa pun. Untuk informasi selengkapnya tentang logika evaluasi ini, lihat Menentukan apakah permintaan ditolak atau diizinkan dalam akun di Panduan Pengguna IAM.

Operasi tingkat objek

Jika objek dimiliki oleh akun pemilik bucket, kebijakan bucket dan kebijakan pengguna IAM akan berfungsi dengan cara yang sama untuk operasi tingkat objek seperti yang mereka lakukan untuk operasi tingkat bucket. Misalnya, jika tidak ada kebijakan bucket, maka bucket secara implisit mengizinkan permintaan objek dari identitas IAM apa pun di akun pemilik bucket. Bucket juga secara implisit menolak permintaan objek dari identitas IAM lainnya dari akun lain, serta menolak permintaan anonim (tidak ditandatangani). Namun, jika tidak ada kebijakan pengguna IAM, peminta mohon (kecuali jika mereka adalah pengguna root) secara implisit ditolak dalam membuat permintaan objek apa pun.

Jika objek dimiliki oleh akun eksternal, maka akses ke objek hanya dapat diberikan melalui daftar kontrol akses (ACL) objek. Kebijakan bucket dan kebijakan pengguna IAM masih dapat digunakan untuk menolak permintaan objek.

Oleh karena itu, untuk memastikan kebijakan bucket atau kebijakan pengguna IAM Anda tidak menyebabkan kesalahan Akses Ditolak (403 Forbidden), pastikan persyaratan berikut terpenuhi:

- Untuk akses akun yang sama, tidak boleh ada pernyataan `Deny` secara eksplisit terhadap peminta yang ingin Anda berikan izin, baik dalam kebijakan bucket maupun kebijakan pengguna IAM. Jika Anda ingin memberikan izin hanya dengan menggunakan kebijakan bucket dan kebijakan pengguna IAM, setidaknya harus ada satu pernyataan `Allow` eksplisit di salah satu kebijakan ini.
- Untuk akses lintas akun, tidak boleh ada pernyataan `Deny` secara eksplisit terhadap peminta yang ingin Anda berikan izin, baik dalam kebijakan bucket maupun kebijakan pengguna IAM. Jika Anda ingin memberikan izin lintas akun hanya dengan menggunakan kebijakan bucket dan kebijakan pengguna IAM, kebijakan bucket dan kebijakan pengguna IAM dari peminta harus menyertakan pernyataan `Allow` eksplisit.

> **Note**
>
> Pernyataan `Allow` dalam kebijakan bucket hanya berlaku untuk objek yang dimiliki oleh akun pemilik bucket yang sama. Namun, pernyataan `Deny` dalam kebijakan bucket berlaku untuk semua objek terlepas dari kepemilikan objek.

Untuk meninjau atau mengedit kebijakan bucket

> **Note**
>
> Untuk melihat atau mengedit kebijakan bucket, Anda harus memiliki izin `s3:GetBucketPolicy`.

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dari daftar Bucket, pilih nama bucket yang ingin Anda lihat atau edit kebijakan bucket-nya.
4. Pilih tab Izin.
5. Di Bawah Kebijakan bucket, pilih Edit. Halaman Edit kebijakan bucket akan muncul.

Untuk meninjau atau mengedit kebijakan bucket Anda menggunakan AWS Command Line Interface (AWS CLI), gunakan get-bucket-policyperintah.

> **Note**
>
> Jika Anda terkunci dari bucket karena kebijakan bucket yang salah, masuk ke AWS Management Console dengan menggunakan kredenal pengguna root Anda. Untuk mendapatkan kembali akses ke bucket Anda, pastikan untuk menghapus kebijakan bucket dengan menggunakan kredensial pengguna root Anda.

Kiat untuk memeriksa izin

Untuk memeriksa apakah peminta memiliki izin yang tepat dalam melakukan operasi Amazon S3, coba hal berikut ini:

- Identifikasi peminta. Jika permintaan tidak ditandatangani, berarti permintaan tersebut merupakan permintaan anonim tanpa kebijakan pengguna IAM. Jika permintaan menggunakan URL yang telah ditandatangani, maka kebijakan pengguna akan sama dengan kebijakan pengguna IAM atau peran yang menandatangani permintaan tersebut.

- Pastikan Anda menggunakan peran atau pengguna IAM yang benar. Anda dapat memverifikasi peran atau pengguna IAM Anda dengan memeriksa sudut kanan atas AWS Management Console atau dengan menggunakan perintah aws sts get-caller-identity.
- Periksa kebijakan IAM yang terkait dengan peran atau pengguna IAM. Gunakan salah satu metode berikut:
  - Uji kebijakan IAM dengan simulator kebijakan IAM.
  - Tinjau berbagai jenis kebijakan IAM yang berbeda.
- Jika perlu, edit kebijakan pengguna IAM Anda.
- Tinjau contoh kebijakan berikut yang secara eksplisit menolak atau mengizinkan akses:
  - Kebijakan mengizinkan pengguna IAM eksplisit: IAM: Mengizinkan dan menolak akses ke beberapa layanan secara terprogram dan di konsol
  - Kebijakan mengizinkan bucket eksplisit: Memberikan izin ke beberapa akun untuk mengunggah objek atau menyetel ACL objek untuk akses publik
  - Tolak kebijakan pengguna IAM secara eksplisit AWS:: Menolak akses AWS berdasarkan permintaan Wilayah AWS
  - Kebijakan bucket menolak eksplisit: Memerlukan SSE-KMS untuk semua objek yang ditulis ke bucket

Pengaturan Amazon S3 ACL

Saat memeriksa setelan ACL, tinjau terlebih dahulu setelan Kepemilikan Objek untuk memeriksa apakah ACL diaktifkan di bucket. Perlu diketahui bahwa izin ACL hanya dapat digunakan untuk memberikan izin dan tidak dapat digunakan untuk menolak permintaan. ACL juga tidak dapat digunakan untuk memberikan akses ke peminta yang ditolak oleh penolakan eksplisit dalam kebijakan bucket atau kebijakan pengguna IAM.

Pengaturan Kepemilikan Objek diatur ke diberlakukan oleh pemilik bucket

Jika pengaturan diberlakukan oleh pemilik bucket diaktifkan, setelan ACL tidak akan menyebabkan kesalahan Akses Ditolak (403 Forbidden) karena setelan ini menonaktifkan semua ACL yang berlaku untuk bucket dan objek. Diberlakukan oleh pemilik bucket adalah setelan default (dan disarankan) untuk bucket Amazon S3.

Pengaturan Kepemilikan Objek diatur ke pilihan pemilik bucket atau penulis objek

Izin ACL masih valid dengan pengaturan pilihan pemilik bucket atau setelan penulis objek. Ada dua jenis ACL: ACL bucket dan ACL objek. Untuk perbedaan antara kedua jenis ACL ini, lihat Pemetaan izin ACL dan izin kebijakan akses.

Bergantung pada tindakan permintaan yang ditolak, periksa izin ACL untuk bucket atau objek:

- Jika Amazon S3 menolak `LIST`, objek `PUT`, `GetBucketAcl`, atau permintaan `PutBucketAcl`, tinjau izin ACL untuk bucket Anda.

  > **Note**
  >
  > Anda tidak dapat memberikan izin objek `GET` dengan pengaturan ACL bucket.

- Jika Amazon S3 menolak permintaan `GET` pada objek S3, atau permintaan PutObjectAcl, maka tinjau izin ACL untuk objek tersebut.

  > **Important**
  >
  > Jika akun pemilik objek berbeda dari akun pemilik bucket, maka akses ke objek tidak dikontrol oleh kebijakan bucket.

Memecahkan masalah kesalahan Akses Ditolak (403 Forbidden) dari permintaan objek `GET` selama kepemilikan objek lintas akun

Tinjau pengaturan Kepemilikan Objek bucket untuk menentukan pemilik objek. Jika Anda memiliki akses ke objek ACL, maka Anda juga dapat memeriksa akun pemilik objek. (Untuk melihat akun pemilik objek, tinjau pengaturan ACL objek di konsol Amazon S3.) Atau, Anda juga dapat membuat permintaan `GetObjectAcl` untuk menemukan ID kanonik pemilik objek untuk memverifikasi akun pemilik objek. Secara default, ACL memberikan izin perizinan eksplisit untuk permintaan `GET` ke akun pemilik objek.

Setelah Anda mengonfirmasi bahwa pemilik objek berbeda dari pemilik bucket, lalu tergantung pada kasus penggunaan dan tingkat akses Anda, pilih salah satu metode berikut untuk membantu mengatasi kesalahan Akses Ditolak (403 Forbidden):

- Nonaktifkan ACL (disarankan)–Metode ini akan berlaku untuk semua objek dan dapat dilakukan oleh pemilik bucket. Metode ini secara otomatis memberikan pemilik bucket kepemilikan dan kontrol penuh atas setiap objek di bucket. Sebelum Anda menerapkan metode ini, periksa prasyarat untuk menonaktifkan ACL. Untuk informasi tentang cara menyetel bucket ke mode diberlakukan oleh pemilik bucket (disarankan), lihat Mengatur Kepemilikan Objek pada bucket yang ada.

  > **Important**
  >
  > Untuk mencegah kesalahan Akses Ditolak (403 Forbidden), pastikan untuk memigrasikan izin ACL ke kebijakan bucket sebelum menonaktifkan ACL. Untuk informasi selengkapnya, lihat Contoh kebijakan bucket untuk migrasi dari izin ACL.

- Ubah pemilik objek menjadi pemilik bucket–Metode ini dapat diterapkan pada objek individual, tetapi hanya pemilik objek (atau pengguna dengan izin yang sesuai) yang dapat mengubah kepemilikan objek. Biaya PUT tambahan mungkin berlaku. (Untuk informasi selengkapnya, lihat harga Amazon S3.) Metode ini memberikan pemilik bucket kepemilikan penuh atas objek, memungkinkan pemilik bucket untuk mengontrol akses ke objek melalui kebijakan bucket.

  Untuk mengubah kepemilikan objek, lakukan salah satu hal berikut:

  - Anda (pemilik bucket) dapat menyalin objek kembali ke bucket.
  - Anda dapat mengubah setelan Kepemilikan Objek bucket menjadi pilihan pemilik bucket. Jika Penentuan Versi dinonaktifkan, objek dalam bucket akan ditimpa. Jika Penentuan Versi diaktifkan, versi duplikat dari objek yang sama akan muncul di bucket, dan pemilik bucket dapat menetapkan aturan siklus hidup yang akan diakhiri. Untuk petunjuk tentang cara mengubah pengaturan Kepemilikan Objek Anda, lihatMenyetel Kepemilikan Objek pada bucket yang ada.

    > **Note**
    >
    > Saat Anda memperbarui setelan Kepemilikan Objek ke plihan pemilik bucket, pengaturan hanya diterapkan ke objek baru yang diunggah ke bucket.

  - Anda dapat meminta pemilik objek mengunggah kembali objek dengan objek ACL terekam `bucket-owner-full-control`.

> **Note**
>
> Untuk unggahan lintas akun, Anda juga dapat meminta ACL objek terekam `bucket-owner-full-control` dalam kebijakan bucket Anda. Untuk contoh kebijakan bucket, lihat Memberikan izin lintas akun untuk mengunggah objek sekaligus memastikan bahwa pemilik bucket memiliki kontrol penuh.

- Simpan penulis objek sebagai pemilik objek-Metode ini tidak mengubah pemilik objek, tetapi memungkinkan Anda untuk memberikan akses ke objek satu per satu. Untuk memberikan akses ke objek, Anda harus memiliki izin `PutObjectAcl` untuk objek tersebut. Kemudian, untuk memperbaiki kesalahan Akses Ditolak (403 Forbidden), tambahkan peminta sebagai penerima untuk mengakses objek di ACL objek. Untuk informasi selengkapnya, lihat Mengonfigurasi ACL.

Pengaturan S3 Block Public Access

Jika permintaan yang gagal melibatkan akses publik atau kebijakan publik, periksa pengaturan S3 Block Public Access di akun, bucket, atau titik akses S3 Anda. Mulai April 2023, semua pengaturan Blok Akses Publik diaktifkan secara default untuk bucket baru. Untuk informasi selengkapnya tentang cara Amazon S3 mendefinisikan “publik,” lihat Arti “publik”.

Saat disetel ke `TRUE`, setelan Blok Akses Publik bertindak sebagai kebijakan penolakan eksplisit yang mengganti izin yang diizinkan oleh ACL, kebijakan bucket, dan kebijakan pengguna IAM. Untuk menentukan apakah setelan Blok Akses Publik menolak permintaan, tinjau skenario berikut:

- Jika daftar kontrol akses (ACL) yang ditentukan bersifat publik, maka pengaturan `BlockPublicAcls` menolak `PutBucketAcl` Anda dan panggilan `PutObjectACL`.
- Jika permintaan menyertakan ACL publik, maka pengaturan `BlockPublicAcls` menolak panggilan `PutObject` Anda.
- Jika pengaturan `BlockPublicAcls` diterapkan ke akun dan permintaan tersebut menyertakan ACL publik, maka setiap panggilan `CreateBucket` yang menyertakan ACL publik akan gagal.
- Jika izin permintaan Anda hanya diberikan oleh ACL publik, maka pengaturan `IgnorePublicAcls` menolak permintaan tersebut.
- Jika kebijakan bucket yang ditentukan memungkinkan akses publik, maka setelan `BlockPublicPolicy` akan menolak panggilan `PutBucketPolicy` Anda.

- Jika pengaturan `BlockPublicPolicy` diterapkan ke titik akses, maka semua panggilan `PutAccessPointPolicy` dan `PutBucketPolicy` yang menentukan kebijakan publik dan dilakukan melalui titik akses akan gagal.
- Jika access point atau bucket memiliki kebijakan publik, maka `RestrictPublicBuckets` pengaturan menolak semua panggilan lintas akun kecuali untuk Layanan AWS prinsipal. Pengaturan ini juga menolak semua panggilan anonim (atau tidak ditandatangani).

Untuk meninjau dan memperbarui konfigurasi setelan Blok Akses Publik, lihatMengonfigurasi pengaturan blokir akses publik untuk bucket S3 Anda.

Pengaturan enkripsi Amazon S3

Amazon S3 mendukung enkripsi di sisi server di bucket Anda. Enkripsi di sisi server adalah enkripsi data di tempat tujuan oleh aplikasi atau layanan yang menerimanya. Amazon S3 mengenkripsi data Anda pada tingkat objek saat menulisnya ke disk di pusat AWS data dan mendekripsi untuk Anda saat Anda mengaksesnya.

Amazon S3 sudah menerapkan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3) sebagai tingkat dasar enkripsi bagi setiap bucket di Amazon S3. Amazon S3 juga memungkinkan Anda menentukan metode enkripsi di sisi server saat mengunggah objek.

Untuk meninjau status enkripsi di sisi server dan setelan enkripsi bucket

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dari daftar Bucket, pilih bucket yang ingin Anda periksa pengaturan enkripsinya.
4. Pilih tab Properti.
5. Gulir ke bawah ke bagian Enkripsi default lalu lihat pengaturan Jenis enkripsi.

Untuk memeriksa pengaturan enkripsi Anda dengan menggunakan AWS CLI, gunakan get-bucket-encryptionperintah.

Untuk memeriksa status enkripsi objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Di panel navigasi kiri, pilih Bucket.
3. Di daftar Bucket, pilih nama bucket yang berisi objek.
4. Dari daftar Objek, pilih nama objek yang ingin ditambahkan atau ubah enkripsinya.

   Halaman detail objek akan muncul.
5. Gulir ke bawah ke bagian Pengaturan enkripsi sisi server untuk melihat pengaturan enkripsi di sisi server objek.

Untuk memeriksa status enkripsi objek Anda dengan menggunakan AWS CLI, gunakan head-objectperintah.

Persyaratan enkripsi dan izin

Amazon S3 mendukung tiga jenis enkripsi di sisi server:

- Enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3)
- Enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS)
- Enkripsi di sisi server dengan kunci yang disediakan pelanggan (SSE-C)

Berdasarkan pengaturan enkripsi Anda, pastikan bahwa persyaratan izin berikut terpenuhi:

- SSE-S3-Tidak diperlukan izin tambahan.
- SSE-KMS (dengan CMK)-Untuk mengunggah objek, izin `kms:GenerateDataKey` pada AWS KMS key diperlukan. Untuk mengunduh objek dan melakukan unggahan objek multibagian, izin `kms:Decrypt` pada kunci KMS diperlukan.
- SSE-KMS (dengan Kunci yang dikelola AWS) — Pemohon harus dari akun yang sama yang memiliki kunci KMS. `aws/s3` Peminta juga harus memiliki izin Amazon S3 yang benar untuk mengakses objek.
- SSE-C (dengan kunci yang disediakan pelanggan)-Tidak diperlukan izin tambahan. Anda dapat mengonfigurasi kebijakan bucket agar memerlukan dan membatasi enkripsi di sisi server dengan kunci enkripsi yang disediakan pelanggan untuk objek di bucket.

Jika objek dienkripsi dengan CMK, pastikan bahwa kebijakan kunci KMS memungkinkan Anda untuk melakukan `kms:GenerateDataKey` atau tindakan `kms:Decrypt`. Untuk petunjuk tentang memeriksa kebijakan kunci KMS Anda, lihat Melihat kebijakan kunci di Panduan Pengembang AWS Key Management Service .

Pengaturan Kunci Objek S3

Jika bucket Anda mengaktifkan Kunci Objek S3 dan objek tersebut dilindungi oleh periode retensi atau penahanan hukum, Amazon S3 akan menampilkan kesalahan Akses Ditolak (403 Forbidden) saat Anda mencoba menghapus objek.

Untuk memeriksa apakah bucket telah mengaktifkan Kunci Objek

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Dari daftar Bucket, pilih nama bucket yang ingin Anda tinjau.
4. Pilih tab Properti.
5. Gulir ke bawah hingga bagian Kunci Objek. Verifikasi apakah pengaturan Kunci Objek Diaktifkan atau Dinonaktifkan.

Untuk menentukan apakah objek dilindungi oleh periode retensi atau penahanan hukum, lihat informasi kunci untuk objek Anda.

Jika objek dilindungi oleh periode retensi atau penahanan hukum, periksa hal-hal berikut:

- Jika versi objek dilindungi oleh mode retensi kepatuhan, tidak ada cara untuk menghapusnya secara permanen. Permintaan `DELETE` permanen dari pemohon, termasuk pengguna root, akan menghasilkan kesalahan Akses Ditolak (403 Forbidden). Selain itu, perlu diketahui bahwa saat Anda mengirimkan permintaan `DELETE` untuk objek yang dilindungi oleh mode retensi kepatuhan, Amazon S3 membuat penanda hapus untuk objek tersebut.
- Jika versi objek dilindungi dengan mode retensi tata kelola dan Anda memiliki izin `s3:BypassGovernanceRetention`, Anda dapat melewati perlindungan dan menghapus versi secara permanen. Untuk informasi selengkapnya, lihat Memintas mode tata kelola.
- Jika versi objek dilindungi oleh penahanan hukum, maka permintaan `DELETE` permanen dapat mengakibatkan kesalahan Akses Ditolak (403 Forbidden). Untuk menghapus versi objek secara permanen, Anda harus menghapus penahanan hukum pada versi objek. Untuk menghapus penahanan hukum, Anda harus memiliki izin `s3:PutObjectLegalHold`. Untuk informasi selengkapnya tentang menghapus penahanan hukum, lihat Mengonfigurasi Kunci Objek S3.

Kebijakan titik akhir VPC

Jika Anda mengakses Amazon S3 menggunakan titik akhir cloud privat virtual (VPC), pastikan kebijakan titik akhir VPC tidak menghalangi Anda untuk mengakses sumber daya Amazon S3. Secara default, kebijakan titik akhir VPC mengizinkan semua permintaan ke Amazon S3. Anda juga dapat mengonfigurasi kebijakan titik akhir VPC untuk membatasi permintaan tertentu. Untuk informasi tentang cara memeriksa kebijakan titik akhir VPC Anda, lihat Mengontrol akses ke titik akhir VPC menggunakan kebijakan titik akhir di Panduan AWS PrivateLink .

AWS Organizations kebijakan

Jika Anda Akun AWS termasuk dalam organisasi, AWS Organizations kebijakan dapat memblokir Anda mengakses sumber daya Amazon S3. Secara default, AWS Organizations kebijakan tidak memblokir permintaan apa pun ke Amazon S3. Namun, pastikan AWS Organizations kebijakan Anda belum dikonfigurasi untuk memblokir akses ke bucket S3. Untuk petunjuk tentang cara memeriksa AWS Organizations kebijakan Anda, lihat Mencantumkan semua kebijakan di Panduan AWS Organizations Pengguna.

Pengaturan titik akses

Jika Anda menerima kesalahan Akses Ditolak (403 Forbidden) saat membuat permintaan melalui Titik Akses Amazon S3, Anda mungkin perlu memeriksa hal-hal berikut:

- Konfigurasi untuk titik akses Anda
- Kebijakan pengguna IAM yang digunakan untuk titik akses Anda
- Kebijakan bucket yang digunakan untuk mengelola atau mengonfigurasi titik akses lintas akun

Konfigurasi dan kebijakan titik akses

- Saat Anda membuat titik akses, Anda dapat memilih untuk menunjuk Internet atau VPC sebagai asal jaringan. Jika asal jaringan disetel ke VPC saja, Amazon S3 akan menolak permintaan apa pun yang dibuat ke titik akses yang tidak berasal dari VPC yang ditentukan. Untuk memeriksa asal jaringan titik akses Anda, lihatMembuat titik akses terbatas pada cloud privat virtual.
- Dengan titik akses, Anda juga dapat mengonfigurasi pengaturan Blok Akses Publik kustom, yang bekerja sama dengan pengaturan Blok Akses Publik di tingkat bucket atau akun. Untuk memeriksa pengaturan Blok Akses Publik kustom Anda, lihat Mengelola akses publik ke titik akses.

- Untuk membuat permintaan yang berhasil ke Amazon S3 dengan menggunakan titik akses, pastikan bahwa peminta memiliki izin IAM yang diperlukan. Untuk informasi selengkapnya, lihat Mengonfigurasi kebijakan IAM untuk menggunakan titik akses.
- Jika permintaan melibatkan titik akses lintas akun, pastikan pemilik bucket telah memperbarui kebijakan bucket untuk mengotorisasi permintaan dari titik akses. Untuk informasi selengkapnya, lihat Memberikan izin untuk titik akses lintas akun.

Jika kesalahan Akses Ditolak (403 Terlarang) masih berlanjut setelah memeriksa semua item dalam topik ini, ambil ID permintaan Amazon S3 Anda dan hubungi untuk panduan tambahan. AWS Support

Memecahkan Masalah Operasi Batch

Daftar topik berikut ini mencantumkan kesalahan umum untuk membantu Anda memecahkan masalah yang mungkin Anda hadapi saat Operasi Batch.

Kesalahan Umum

- Laporan pekerjaan tidak terkirim saat ada masalah izin atau mode retensi Kunci Objek S3 diaktifkan
- Kegagalan Replikasi Batch S3 dengan kesalahan: Pembuatan manifes tidak menemukan kunci yang cocok dengan kriteria filter
- Kegagalan Operasi Batch terjadi setelah menambahkan aturan replikasi baru ke konfigurasi replikasi yang ada
- Operasi Batch gagal objek dengan kesalahan 400 InvalidRequest: Tugas gagal karena hilang VersionId
- Buat kegagalan pekerjaan dengan opsi tag pekerjaan diaktifkan
- Akses Ditolak untuk membaca manifes

Laporan pekerjaan tidak terkirim saat ada masalah izin atau mode retensi Kunci Objek S3 diaktifkan

Kesalahan berikut terjadi jika izin yang diperlukan tidak ada atau mode retensi Kunci Objek (baik mode tata kelola atau mode kepatuhan) diaktifkan di bucket tujuan.

Kesalahan: Alasan kegagalan. Laporan pekerjaan tidak dapat ditulis ke bucket laporan Anda. Silakan periksa izin Anda.

Peran IAM dan kebijakan kepercayaan harus dikonfigurasi agar Operasi Batch S3 mengakses objek PUT di bucket tempat laporan akan dikirimkan. Jika izin yang diperlukan ini hilang, akan terjadi kegagalan pengiriman laporan pekerjaan.

Saat mode retensi diaktifkan, bucket dilindungi write-once-read-many (WORM). Kunci Objek dengan mode retensi yang diaktifkan pada bucket tujuan tidak didukung sehingga upaya pengiriman laporan penyelesaian pekerjaan gagal. Untuk memperbaiki masalah ini, pilih bucket tujuan untuk laporan penyelesaian pekerjaan yang tidak mengaktifkan mode retensi Kunci Objek.

Kegagalan Replikasi Batch S3 dengan kesalahan: Pembuatan manifes tidak menemukan kunci yang cocok dengan kriteria filter

Kesalahan: Pembuatan manifes tidak menemukan kunci yang cocok dengan kriteria filter.

Kesalahan ini terjadi untuk salah satu reasi berikut:

- Saat objek di bucket sumber disimpan di kelas penyimpanan S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive.

  Untuk menggunakan Replikasi Batch pada objek-objek ini, pertama-tama kembalikan ke kelas penyimpanan Standar S3 dengan menggunakan operasi S3 Initiate Restore Object dalam pekerjaan Operasi Batch. Untuk informasi selengkapnya, lihat Memulihkan objek yang diarsipkan dan Memulihkan objek (Operasi Batch). Setelah Anda memulihkan objek, Anda dapat mereplikasi mereka dengan menggunakan pekerjaan Replikasi Batch.
- Jika kriteria filter yang diberikan tidak cocok dengan objek yang valid di bucket sumber.

  Verifikasi dan perbaiki kriteria filter. Misalnya, dalam aturan Replikasi Batch, kriteria filter mencari semua objek di *`example-s3-bucket`* dengan awalan. `Tax/` Jika nama awalan dimasukkan secara tidak akurat, dengan garis miring di awal dan akhir, `/Tax/` bukan hanya di akhir, maka tidak ada objek S3 yang ditemukan. Untuk mengatasi kesalahan, perbaiki awalan, dalam hal ini, dari `/Tax/` ke `Tax/` dalam aturan replikasi.

Kegagalan Operasi Batch terjadi setelah menambahkan aturan replikasi baru ke konfigurasi replikasi yang ada

Operasi Batch mencoba melakukan replikasi objek yang ada untuk setiap aturan dalam konfigurasi replikasi bucket sumber. Jika ada masalah dengan salah satu aturan replikasi yang ada, kegagalan dapat terjadi.

Laporan penyelesaian pekerjaan Operasi Batch menjelaskan alasan kegagalan pekerjaan. Untuk daftar kesalahan umum, lihat Penyebab kegagalan replikasi Amazon S3.

Operasi Batch gagal objek dengan kesalahan 400 InvalidRequest: Tugas gagal karena hilang VersionId

Contoh kesalahan berikut terjadi jika pekerjaan Operasi Batch melakukan tindakan pada objek dalam bucket Penentuan Versi dan menemukan objek dalam manifes dengan bidang ID versi kosong.

Kesalahan: *`BUCKET_NAME, awalan/file_name, gagal,400`*,, Tugas gagal karena hilang InvalidRequest VersionId

Kesalahan ini terjadi karena bidang ID versi dalam manifes adalah string kosong, bukan string `null` literal.

Operasi Batch akan gagal untuk objek atau banyak objek tertentu, tetapi tidak seluruh pekerjaan. Masalah ini terjadi jika format manifes dikonfigurasi untuk menggunakan ID versi selama operasi. Pekerjaan non-versi tidak mengalami masalah ini karena mereka hanya beroperasi pada versi terbaru dari setiap objek dan mengabaikan ID versi dalam manifes.

Untuk memperbaiki masalah ini, ubah ID versi kosong menjadi string `null`. Untuk informasi selengkapnya, lihat the section called “Mengkonversi string ID versi kosong ke string null”.

Buat kegagalan pekerjaan dengan opsi tag pekerjaan diaktifkan

Tanpa izin `s3:PutJobTagging`, membuat pekerjaan Operasi Batch dengan opsi tag pekerjaan diaktifkan menyebabkan kesalahan `403 access denied`.

Untuk membuat pekerjaan Operasi Batch dengan opsi tag pekerjaan diaktifkan, pengguna AWS Identity and Access Management (IAM) yang membuat pekerjaan Operasi Batch harus memiliki `s3:PutJobTagging` izin selain `s3:CreateJob` izin.

Untuk informasi lebih lanjut tentang izin yang diperlukan untuk Operasi Batch, lihatthe section called “Memberikan izin”.

Akses Ditolak untuk membaca manifes

Jika Operasi Batch tidak dapat membaca file manifes saat Anda mencoba membuat pekerjaan Operasi Batch, kesalahan berikut dapat terjadi.

AWS CLI

Alasan kegagalan Membaca manifes dilarang: AccessDenied

Konsol Amazon S3

Peringatan: Tidak bisa mendapatkan ETag objek manifes. Tentukan objek yang berbeda untuk melanjutkan.

Untuk mengatasi masalah ini, lakukan hal berikut:

- Verifikasi bahwa peran IAM untuk Akun AWS yang Anda gunakan untuk membuat pekerjaan Operasi Batch memiliki `s3:GetObject` izin. Peran IAM akun harus memiliki izin `s3:GetObject` untuk mengizinkan Operasi Batch membaca file manifes.

  Untuk informasi lebih lanjut tentang izin yang diperlukan untuk Operasi Batch, lihat the section called “Memberikan izin”.
- Periksa metadata objek manifes untuk ketidakcocokan akses apa pun dengan Kepemilikan Objek S3. Untuk informasi selengkapnya tentang Kepemilikan Objek S3, lihat the section called “Mengontrol kepemilikan objek”.
- Periksa apakah kunci AWS Key Management Service (AWS KMS) digunakan untuk mengenkripsi file manifes.

  Operasi Batch mendukung laporan inventaris CSV yang dienkripsi AWS KMS. Namun, Operasi Batch tidak mendukung file manifes CSV yang dienkripsi AWS KMS. Lihat informasi yang lebih lengkap di Mengonfigurasi Inventaris Amazon S3 dan Menentukan manifes.

Penyelesaian masalah CORS

Jika Anda mengalami perilaku yang tidak terduga saat mengakses bucket yang ditetapkan dengan konfigurasi CORS, coba langkah-langkah berikut untuk memecahkan masalah:

1. Verifikasi bahwa konfigurasi CORS diatur pada bucket.

   Jika konfigurasi CORS diatur, konsol menampilkan tautan Edit Konfigurasi CORS di bagian Izin dari bucket Properti.
2. Catat permintaan dan respons lengkap menggunakan alat pilihan Anda. Untuk setiap permintaan yang diterima Amazon S3, harus ada aturan CORS yang sesuai dengan data dalam permintaan Anda, sebagai berikut:

a. Verifikasi bahwa permintaan memiliki header Asal.

   Jika header tidak ada, Amazon S3 tidak memperlakukan permintaan tersebut sebagai permintaan lintas asal, dan tidak mengirimkan header respons CORS dalam respons.

b. Verifikasi bahwa header Asal dalam permintaan Anda cocok dengan setidaknya satu dari elemen `AllowedOrigin` dalam `CORSRule`.

   Skema, host, dan nilai port di header permintaan Asal harus sesuai elemen `AllowedOrigin` di `CORSRule`. Sebagai contoh, jika Anda mengatur `CORSRule` untuk mengizinkan asal `http://www.example.com`, maka asal `https://www.example.com` dan `http://www.example.com:80` dalam permintaan Anda tidak cocok dengan asal yang diizinkan dalam konfigurasi Anda.

c. Verifikasi bahwa metode dalam permintaan Anda (atau dalam permintaan preflight, metode yang ditentukan dalam `Access-Control-Request-Method`) adalah salah satu elemen `AllowedMethod` di `CORSRule` yang sama.

d. Untuk permintaan preflight, jika permintaan mencakup header `Access-Control-Request-Headers`, verifikasikan bahwa `CORSRule` termasuk entri `AllowedHeader` untuk setiap nilai dalam header `Access-Control-Request-Headers`.

Memecahkan masalah Siklus Hidup S3

Informasi berikut dapat membantu Anda memecahkan masalah umum pada aturan Siklus Hidup S3.

Topik

- Saya menjalankan operasi daftar di bucket saya dan melihat objek yang menurut saya sudah kedaluwarsa atau ditransisikan oleh aturan siklus hidup.
- Bagaimana cara memantau tindakan yang diambil oleh aturan siklus hidup saya?
- Jumlah objek S3 saya masih meningkat, bahkan setelah menyiapkan aturan siklus hidup pada bucket yang mendukung versi.
- Bagaimana cara mengosongkan bucket S3 saya dengan menggunakan aturan siklus hidup?
- Tagihan Amazon S3 saya meningkat setelah mentransisikan objek ke kelas penyimpanan berbiaya lebih rendah.
- Saya telah memperbarui kebijakan bucket saya, tetapi objek S3 saya masih dihapus oleh aturan siklus hidup yang kedaluwarsa.
- Dapatkah saya memulihkan objek S3 yang dikahiri oleh aturan Siklus Hidup S3?

Saya menjalankan operasi daftar di bucket saya dan melihat objek yang menurut saya sudah kedaluwarsa atau ditransisikan oleh aturan siklus hidup.

Transisi objek Siklus Hidup S3 dan waktu kedaluwarsa objek adalah operasi asinkron. Oleh karena itu, dapat terjadi penundaan antara waktu objek memenuhi syarat untuk kedaluwarsa atau transisi dan waktu ketika objek tersebut benar-benar ditransisikan atau kedaluwarsa. Perubahan dalam penagihan diterapkan segera setelah aturan siklus hidup terpenuhi, bahkan jika tindakan tidak selesai. Pengecualian untuk perilaku ini adalah jika Anda memiliki aturan siklus hidup yang disetel ke kelas penyimpanan S3 Intelligent-Tiering. Dalam hal ini, perubahan penagihan tidak terjadi hingga objek telah ditransisikan ke S3 Intelligent-Tiering. Untuk informasi selengkapnya tentang perubahan dalam penagihan, lihat Menyetel konfigurasi siklus hidup pada bucket.

> **Note**
>
> Amazon S3 tidak mentransisikan objek yang lebih kecil dari 128 KB dari kelas penyimpanan S3 Standard atau S3 Standard-IA ke kelas penyimpanan S3 Intelligent-Tiering, S3 Standard-IA, atau S3 One Zone-IA.

Bagaimana cara memantau tindakan yang diambil oleh aturan siklus hidup saya?

Untuk memantau tindakan yang diambil oleh aturan siklus hidup, Anda dapat menggunakan fitur berikut:

- Pemberitahuan Acara S3 - Anda dapat mengatur Pemberitahuan Acara S3 sehingga Anda diberi tahu tentang kedaluwarsa atau peristiwa transisi Siklus Hidup S3 apa pun.
- Log akses server S3 — Anda dapat mengaktifkan log akses server untuk bucket S3 Anda untuk menangkap tindakan Siklus Hidup S3, seperti transisi objek ke kelas penyimpanan lain atau kedaluwarsa objek. Untuk informasi lebih lanjut, lihat Siklus hidup dan pencatatan.

Untuk melihat perubahan dalam penyimpanan yang disebabkan oleh tindakan siklus hidup setiap hari, sebaiknya gunakan dasbor Lensa Penyimpanan S3 alih-alih menggunakan metrik Amazon. CloudWatch Di dasbor Lensa Penyimpanan, Anda dapat melihat metrik berikut, yang memantau jumlah atau ukuran objek:

- Byte versi terkini
- Jumlah objek versi saat ini
- Byte bukan versi terkini
- Jumlah objek versi tidak terkini
- Hapus Jumlah Objek Penanda
- Hapus byte penyimpanan penanda
- Byte unggahan multibagian yang tidak lengkap
- Jumlah objek unggahan multibagian yang tidak lengkap

Jumlah objek S3 saya masih meningkat, bahkan setelah menyiapkan aturan siklus hidup pada bucket yang mendukung versi.

Dalam bucket berkemampuan versi, saat objek kedaluwarsa, objek tersebut tidak sepenuhnya dihapus dari bucket. Sebagai gantinya, penanda hapus dibuat sebagai versi terbaru dari objek. Hapus penanda masih dihitung sebagai objek. Oleh karena itu, jika aturan siklus hidup dibuat agar hanya mengakhiri versi saat ini, maka jumlah objek di bucket S3 sebenarnya akan meningkat dan bukan turun.

Misalnya, bucket S3 dengan Penentuan Versi yang diaktifkan memiliki 100 objek, dan aturan siklus hidup diatur agar mengakhiri versi objek saat ini setelah 7 hari. Setelah hari ketujuh, jumlah objek meningkat menjadi 200 karena 100 penanda hapus dibuat selain dari 100 objek yang asli, yang sekarang menjadi versi tidak terkini. Untuk informasi selengkapnya tentang tindakan aturan konfigurasi Siklus Hidup S3 untuk bucket dengan Penentuan Versi diaktifkan, lihat Mengatur konfigurasi siklus hidup pada bucket.

Untuk menghapus objek secara permanen, tambahkan konfigurasi siklus hidup tambahan untuk menghapus versi objek sebelumnya, penanda hapus kedaluwarsa, dan unggahan multibagian yang tidak lengkap. Untuk petunjuk tentang cara membuat aturan siklus hidup baru, lihat Mengatur konfigurasi siklus hidup pada bucket.

> **Note**
>
> - Amazon S3 membulatkan transisi atau tanggal kedaluwarsa suatu objek ke tengah malam UTC pada hari berikutnya.

Saat mengevaluasi objek untuk tindakan siklus hidup, Amazon S3 menggunakan waktu pembuatan objek di UTC. Misalnya, pertimbangkan bucket nonversioned dengan aturan siklus hidup yang dikonfigurasi untuk objek kedaluwarsa setelah satu hari. Misalkan sebuah objek dibuat pada 1 Januari pukul 17:05 Pacific Daylight Time (PDT), yang sesuai dengan 2 Januari pukul 00:05 UTC. Objek menjadi berumur satu hari pada 00:05 UTC pada 3 Januari, yang membuatnya memenuhi syarat untuk kedaluwarsa ketika S3 Lifecycle mengevaluasi objek pada 00:00 UTC pada 4 Januari.

Karena tindakan siklus hidup Amazon S3 terjadi secara asinkron, mungkin ada beberapa penundaan antara tanggal yang ditentukan dalam aturan siklus hidup dan transisi fisik objek yang sebenarnya. Untuk informasi selengkapnya, lihat Transisi atau penundaan kedaluwarsa.

Untuk informasi lebih lanjut, lihat Aturan siklus hidup: Berdasarkan usia objek.

- Untuk objek S3 yang dilindungi oleh Kunci Objek, versi saat ini tidak dihapus secara permanen. Sebagai gantinya, penanda hapus ditambahkan ke objek, membuatnya menjadi tidak terkini. Versi tidak terkini kemudian dipertahankan dan tidak kedaluwarsa secara permanen.

Bagaimana cara mengosongkan bucket S3 saya dengan menggunakan aturan siklus hidup?

Aturan Siklus Hidup S3 adalah alat yang efektif untuk mengosongkan bucket S3 dengan jutaan objek. Untuk menghapus sejumlah besar objek dari bucket S3 Anda, pastikan untuk menggunakan dua pasang aturan siklus hidup ini:

- Akhiri versi objek saat ini dan Hapus secara permanen versi objek sebelumnya
- Hapus penanda hapus kedaluwarsa dan Hapus unggahan multibagian yang tidak lengkap

Untuk langkah-langkah cara membuat aturan konfigurasi siklus hidup baru, lihat Mengatur konfigurasi siklus hidup pada bucket.

> **Note**
>
> Untuk objek S3 yang dilindungi oleh Kunci Objek, versi saat ini tidak dihapus secara permanen. Sebagai gantinya, penanda hapus ditambahkan ke objek, membuatnya menjadi tidak terkini. Versi tidak terkini kemudian dipertahankan dan tidak kedaluwarsa secara permanen.

Tagihan Amazon S3 saya meningkat setelah mentransisikan objek ke kelas penyimpanan berbiaya lebih rendah.

Ada beberapa alasan mengapa tagihan Anda mungkin meningkat setelah mentransisikan objek ke kelas penyimpanan berbiaya lebih rendah:

- Biaya overhead S3 Glacier untuk objek-objek kecil

  Untuk setiap objek yang ditransisikan ke S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive, total overhead sebesar 40 KB akan dikaitkan dengan pembaruan penyimpanan ini. Sebagai bagian dari overhead 40 KB, 8 KB digunakan untuk menyimpan metadata dan nama objek. 8 KB ini dibebankan sesuai dengan tarif Standar S3. 32 KB sisanya digunakan untuk pengindeksan dan metadata terkait. 32 KB ini dikenakan biaya sesuai dengan harga S3 Glacier Flexible Retrieval atau S3 Glacier Deep Archive.

  Oleh karena itu, jika Anda menyimpan banyak objek berukuran lebih kecil, kami tidak menyarankan menggunakan transisi siklus hidup. Sebagai gantinya, untuk mengurangi biaya tambahan, pertimbangkan mengumpulkan banyak objek kecil menjadi lebih kecil sebelum menyimpannya di Amazon S3. Untuk informasi lebih lanjut tentang pertimbangan biaya, lihat Transisi ke kelas penyimpanan (pengarsipan objek) S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive.
- Biaya penyimpanan minimum

  Beberapa kelas penyimpanan S3 memiliki persyaratan durasi penyimpanan minimum. Objek yang dihapus, ditimpa, atau ditransisikan dari kelas-kelas tersebut sebelum durasi minimum dipenuhi dikenakan biaya transisi awal atau penghapusan prorata. Persyaratan durasi penyimpanan minimum ini adalah sebagai berikut:
  - S3 Standard-IA dan S3 One Zone-IA-30 hari
  - S3 Glacier Flexible Retrieval dan S3 Glacier Instant Retrieval-90 hari
  - S3 Glacier Deep Archive-180 hari

Untuk informasi selengkapnya tentang persyaratan ini, lihat bagian Batasan dari Objek transisi menggunakan Siklus Hidup S3. Untuk informasi harga S3 umum, lihat harga Amazon S3 dan Kalkulator Harga AWS.

- Biaya transisi siklus hidup

  Setiap kali objek ditransisikan ke kelas penyimpanan yang berbeda berdasarkan aturan siklus hidup, Amazon S3 menghitung transisi tersebut sebagai satu permintaan transisi. Biaya untuk permintaan transisi ini merupakan tambahan dari biaya kelas penyimpanan ini. Jika Anda berencana untuk mentransisi sejumlah besar objek, pertimbangkan biaya permintaan saat melakukan transisi ke tingkat yang lebih rendah. Untuk informasi selengkapnya, lihat Harga Amazon S3.

Saya telah memperbarui kebijakan bucket saya, tetapi objek S3 saya masih dihapus oleh aturan siklus hidup yang kedaluwarsa.

Pernyataan `Deny` dalam kebijakan bucket tidak mencegah berakhirnya objek yang ditentukan dalam aturan siklus hidup. Tindakan siklus hidup (seperti transisi atau kedaluwarsa) tidak menggunakan operasi `DeleteObject` S3. Sebagai gantinya, tindakan Siklus Hidup S3 dilakukan dengan menggunakan titik akhir S3 internal. (Untuk informasi lebih lanjut, lihat Siklus hidup dan pencatatan.)

Untuk mencegah aturan siklus hidup Anda mengambil tindakan apa pun, Anda harus mengedit, menghapus, atau menonaktifkan aturan tersebut.

Dapatkah saya memulihkan objek S3 yang dikahiri oleh aturan Siklus Hidup S3?

Satu-satunya cara untuk memulihkan objek yang diakhiri oleh Siklus Hidup S3 adalah melalui Penentuan Versi, yang harus dilakukan sebelum objek memenuhi syarat untuk kedaluwarsa. Anda tidak dapat membatalkan operasi kedaluwarsa yang dilakukan oleh aturan siklus hidup. Jika objek dihapus secara permanen oleh aturan Siklus Hidup S3 yang ada, Anda tidak dapat memulihkan objek ini. Untuk mengaktifkan Penentuan Versi pada bucket, lihat the section called “Menggunakan Penentuan Versi S3”.

Jika Anda telah menerapkan Penentuan Versi ke bucket dan versi objek yang tidak terkini masih utuh, Anda dapat memulihkan versi sebelumnya dari objek yang kedaluwarsa. Untuk informasi selengkapnya tentang perilaku tindakan aturan Siklus Hidup S3 dan status Penentuan Versi, lihat

tabel status aksi Siklus Hidup dan Penentuan Versi bucket di Elemen untuk menjelaskan tindakan siklus hidup.

> **Note**
>
> Jika bucket S3 dilindungi oleh AWS Backup atau Replikasi S3, Anda mungkin juga dapat menggunakan fitur ini untuk memulihkan objek yang kedaluwarsa.

Memecahkan masalah replikasi

Bagian ini mencantumkan tips pemecahan masalah untuk Replikasi Amazon S3 dan informasi tentang kesalahan Replikasi Batch S3.

Topik

- Kiat pemecahan masalah untuk Replikasi S3
- Kesalahan Replikasi Batch

Kiat pemecahan masalah untuk Replikasi S3

Jika replika objek tidak muncul di bucket tujuan setelah Anda mengonfigurasi replikasi, gunakan kiat pemecahan masalah ini untuk mengidentifikasi dan memperbaiki masalah.

- Mayoritas objek bereplikasi dalam 15 menit. Waktu yang dibutuhkan Amazon S3 untuk mereplikasi suatu objek tergantung pada beberapa faktor, termasuk pasangan Wilayah sumber dan tujuan, serta ukuran objek. Untuk objek besar, replikasi dapat memakan waktu hingga beberapa jam. Untuk visibilitas waktu replikasi, Anda dapat menggunakan Kontrol Waktu Replikasi S3 (S3 RTC).

  Jika objek yang direplikasi berukuran besar, tunggu beberapa saat sebelum memeriksa apakah objek tersebut muncul di tempat tujuan. Anda juga dapat memeriksa status replikasi objek sumber. Jika status replikasi objek adalah `PENDING`, Amazon S3 belum menyelesaikan replikasi. Jika status replikasi objek adalah `FAILED`, periksa konfigurasi replikasi yang ditetapkan pada bucket sumber. Selain itu, untuk menerima informasi tentang kegagalan selama replikasi, Anda dapat mengatur replikasi Notifikasi Peristiwa Amazon S3 agar menerima peristiwa kegagalan. Untuk informasi lebih lanjut, lihat Menerima peristiwa kegagalan replikasi dengan Notifikasi Peristiwa Amazon S3.
- Anda dapat memanggil operasi `HeadObject` API untuk memeriksa status replikasi suatu objek. Operasi `HeadObject` API mengembalikan `PENDING`,`COMPLETED`, atau status `FAILED` replikasi

objek. Sebagai respons terhadap panggilan API `HeadObject`, status replikasi dikembalikan dalam elemen `x-amz-replication-status`.

> **Note**
>
> Untuk menjalankan `HeadObject`, Anda harus memiliki akses baca ke objek yang Anda minta. Permintaan `HEAD` memiliki opsi yang sama dengan permintaan `GET`, tanpa melakukan operasi `GET`. Misalnya, untuk menjalankan permintaan `HeadObject` dengan menggunakan AWS Command Line Interface (AWS CLI), Anda dapat menjalankan perintah berikut. Ganti *`user input placeholders`* dengan informasi Anda sendiri.
>
> ```
> aws s3api head-object --bucket my-bucket --key index.html
> ```

- Setelah `HeadObject` mengembalikan objek dengan status replikasi `FAILED`, Anda dapat menggunakan Replikasi Batch S3 untuk mereplikasi objek yang gagal tersebut. Atau, Anda dapat mengunggah ulang objek yang gagal ke bucket sumber, yang akan memulai replikasi untuk objek baru.
- Dalam konfigurasi replikasi pada bucket sumber, verifikasi hal berikut:
  - Amazon Resource Name (ARN) dari bucket tujuan sudah benar.
  - Prefiks nama kunci sudah benar. Misalnya, jika Anda mengatur konfigurasi agar mereplikasi objek dengan awalan `Tax`, hanya objek dengan nama kunci seperti `Tax/document1` atau `Tax/document2` yang direplikasi. Objek dengan nama kunci `document3` tidak direplikasi.
  - Status dari aturan replikasi adalah `Enabled`.
- Verifikasi bahwa versioning belum ditangguhkan pada bucket mana pun dalam konfigurasi replikasi. Bucket sumber dan tujuan harus memiliki versioning yang diaktifkan.
- Jika aturan replikasi disetel ke Ubah kepemilikan objek ke pemilik bucket tujuan, maka peran AWS Identity and Access Management (IAM) yang digunakan untuk replikasi harus memiliki izin `s3:ObjectOwnerOverrideToBucketOwner`. Izin ini diberikan pada sumber daya (dalam hal ini, bucket tujuan). Misalnya, pernyataan `Resource` berikut menunjukkan cara memberikan izin ini di bucket tujuan:

```
{
  "Effect":"Allow",
  "Action":[
    "s3:ObjectOwnerOverrideToBucketOwner"
  ],
```

```
  "Resource":"arn:aws:s3:::DestinationBucket/*"
}
```

- Jika bucket tujuan dimiliki oleh akun lain, pemilik bucket tujuan juga harus memberikan izin `s3:ObjectOwnerOverrideToBucketOwner` kepada pemilik bucket sumber melalui kebijakan bucket tujuan. Untuk menggunakan kebijakan bucket contoh berikut, ganti *`user input placeholders`* dengan informasi Anda sendiri:

```
{
  "Version": "2012-10-17",
  "Id": "Policy1644945280205",
  "Statement": [
    {
      "Sid": "Stmt1644945277847",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789101:role/s3-replication-role"
      },
      "Action": [
        "s3:ReplicateObject",
        "s3:ReplicateTags",
        "s3:ObjectOwnerOverrideToBucketOwner"
      ],
      "Resource": "arn:aws:s3:::DestinationBucket/*"
    }
  ]
}
```

> **Note**
>
> Jika setelan kepemilikan objek bucket tujuan menyertakan Diberlakukan oleh pemilik bucket, maka Anda tidak perlu memperbarui setelan ke Ubah kepemilikan objek ke pemilik bucket tujuan dalam aturan replikasi. Perubahan kepemilikan objek akan terjadi secara default. Untuk informasi selengkapnya tentang mengubah kepemilikan replika, lihat Mengubah pemilik replika.

- Jika Anda menyetel konfigurasi replikasi dalam skenario lintas akun, di mana bucket sumber dan tujuan dimiliki oleh yang berbedaAkun AWS, bucket tujuan tidak dapat dikonfigurasi sebagai bucket Requester Pays. Untuk informasi selengkapnya, lihat Menggunakan bucket Pembayaran Pemohon untuk transfer dan penggunaan penyimpanan.

- Jika objek sumber bucket dienkripsi dengan kunci AWS Key Management Service (AWS KMS), maka aturan replikasi harus dikonfigurasi untuk menyertakan objek yang dienkripsi AWS KMS. Pastikan untuk memilih Replikasi objek yang dienkripsi AWS KMS di bawah pengaturan Enkripsi Anda di konsol Amazon S3. Kemudian, pilih kunci AWS KMS untuk mengenkripsi objek tujuan.

> **Note**
>
> Jika bucket tujuan berada di akun yang berbeda, tentukan kunci yang dikelola pelanggan AWS KMS yang dimiliki oleh akun tujuan. Jangan gunakan kunci terkelola Amazon S3 default (`aws/s3`). Menggunakan kunci default akan mengenkripsi objek dengan kunci terkelola Amazon S3 yang dimiliki oleh akun sumber, sehingga mencegah objek dibagikan dengan akun lain. Akibatnya, akun tujuan tidak akan dapat mengakses objek di bucket tujuan.

Untuk menggunakan kunci AWS KMS milik akun tujuan untuk mengenkripsi objek tujuan, akun tujuan harus memberikan izin `kms:GenerateDataKey` dan `kms:Encrypt` untuk peran replikasi dalam kebijakan kunci KMS. Untuk menggunakan contoh pernyataan berikut ini dalam kebijakan kunci KMS Anda, ganti *`user input placeholders`* dengan informasi Anda sendiri:

```
{
    "Sid": "AllowS3ReplicationSourceRoleToUseTheKey",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::123456789101:role/s3-replication-role"
    },
    "Action": ["kms:GenerateDataKey", "kms:Encrypt"],
    "Resource": "*"
}
```

Jika Anda menggunakan tanda bintang (`*`) untuk pernyataan `Resource` dalam kebijakan kunci AWS KMS, kebijakan akan memberikan izin untuk menggunakan kunci KMS hanya pada peran replikasi. Kebijakan ini tidak mengizinkan peran replikasi untuk meningkatkan izinnya.

Secara default, kebijakan kunci KMS memberikan izin penuh kepada pengguna root ke kunci tersebut. Izin ini dapat didelegasikan ke pengguna lain di akun yang sama. Kecuali terdapat pernyataan `Deny` dalam kebijakan kunci KMS sumber, menggunakan kebijakan IAM untuk memberikan izin peran replikasi ke kunci KMS sumber sudah cukup.

> **Note**
>
> Kebijakan kunci KMS yang membatasi akses ke rentang CIDR tertentu, titik akhir VPC, atau titik akses S3 dapat menyebabkan replikasi gagal.

Jika kunci KMS sumber atau tujuan memberikan izin berdasarkan konteks enkripsi, konfirmasikan bahwa Kunci Bucket Amazon S3 diaktifkan untuk bucket. Jika bucket mengaktifkan Kunci Bucket S3, konteks enkripsi harus berupa sumber daya tingkat bucket, seperti ini:

```
"kms:EncryptionContext:arn:aws:arn": [
     "arn:aws:s3:::SOURCE_BUCKET_NAME"
     ]
"kms:EncryptionContext:arn:aws:arn": [
     "arn:aws:s3:::DESTINATION_BUCKET_NAME"
     ]
```

Selain izin yang diberikan oleh kebijakan kunci KMS, akun sumber harus menambahkan izin minimum berikut ke kebijakan IAM peran replikasi:

```
{
    "Effect": "Allow",
    "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
    ],
    "Resource": [
        "SourceKmsKeyArn"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "kms:GenerateDataKey",
        "kms:Encrypt"
    ],
    "Resource": [
        "DestinationKmsKeyArn"
    ]
}
```

Untuk informasi selengkapnya tentang cara mereplikasi objek yang dienkripsi dengan AWS KMS, lihat Mereplikasi objek terenkripsi.

- Jika bucket tujuan dimiliki oleh Akun AWS lain, verifikasi bahwa pemilik bucket memiliki kebijakan bucket tujuan yang memungkinkan pemilik bucket sumber mereplikasi objek. Sebagai contoh, silakan lihat Mengonfigurasi replikasi ketika bucket sumber dan tujuan dimiliki oleh akun yang berbeda.
- Jika objek Anda masih tidak bereplikasi setelah Anda memvalidasi izin, periksa pernyataan `Deny` eksplisit di lokasi berikut:
  - Pernyataan `Deny` dalam kebijakan bucket sumber atau tujuan. Replikasi gagal jika kebijakan bucket menghalangi akses ke peran replikasi untuk tindakan-tindakan berikut:

    Bucket sumber:

    ```
    "s3:GetReplicationConfiguration",
    "s3:ListBucket",
    "s3:GetObjectVersionForReplication",
    "s3:GetObjectVersionAcl",
    "s3:GetObjectVersionTagging"
    ```

    Bucket tujuan:

    ```
    "s3:ReplicateObject",
    "s3:ReplicateDelete",
    "s3:ReplicateTags"
    ```

  - Pernyataan atau batas izin `Deny` yang dilampirkan pada peran IAM dapat menyebabkan replikasi gagal.
  - Pernyataan `Deny` dalam kebijakan kontrol layanan AWS Organizations yang dilampirkan pada akun sumber atau tujuan dapat menyebabkan replikasi gagal.
- Jika replika objek tidak muncul di bucket tujuan, masalah berikut mungkin dapat mencegah replikasi:

  - Amazon S3 tidak mereplikasi objek dalam bucket sumber yang merupakan replika yang dibuat oleh konfigurasi replikasi lain. Misalnya, jika Anda mengatur konfigurasi replikasi dari bucket A ke bucket B ke bucket C, Amazon S3 tidak mereplikasi replika objek di bucket B ke bucket C.
  - Pemilik bucket sumber dapat memberikan izin Akun AWS lainnya untuk mengunggah objek. Secara default, pemilik bucket sumber tidak memiliki izin untuk objek yang dibuat oleh akun lain. Konfigurasi replikasi hanya mereplikasi objek yang izin aksesnya dimiliki pemilik bucket sumber. Pemilik bucket sumber dapat memberikan izin Akun AWS lainnya untuk membuat objek secara kondisional, yang memerlukan izin akses eksplisit pada objek tersebut. Untuk contoh kebijakan, lihat Berikan izin lintas akun untuk unggah objek sekaligus memastikan bahwa pemilik bucket memiliki kendali penuh.
- Misalkan bahwa dalam konfigurasi replikasi, Anda menambahkan aturan untuk mereplikasi subset objek yang memiliki tag tertentu. Dalam kasus ini, Anda harus menetapkan kunci dan nilai tag spesifik pada saat objek dibuat agar Amazon S3 mereplikasi objek. Jika Anda membuat objek terlebih dahulu lalu menambahkan tag ke objek yang sudah ada, Amazon S3 tidak akan mereplikasi objek.
- Gunakan Notifikasi Peristiwa Amazon S3 untuk memberi tahu Anda tentang instans saat objek tidak bereplikasi ke Wilayah AWS tujuannya. Notifikasi peristiwa Amazon S3 tersedia melalui Amazon Simple Queue Service (Amazon SQS), Amazon Simple Notification Service (Amazon SNS), atau AWS Lambda. Untuk informasi selengkapnya, lihat Menerima peristiwa kegagalan replikasi dengan Notifikasi Peristiwa Amazon S3.

  Anda juga dapat melihat alasan kegagalan replikasi dengan menggunakan Notifikasi Peristiwa Amazon S3. Untuk meninjau daftar alasan kegagalan, lihat Alasan kegagalan replikasi Amazon S3.

Kesalahan Replikasi Batch

Untuk memecahkan masalah objek yang tidak bereplikasi ke bucket tujuan, periksa berbagai jenis izin untuk bucket, peran replikasi, dan peran IAM yang digunakan untuk membuat pekerjaan Replikasi Batch. Selain itu, pastikan untuk memeriksa pengaturan akses publik dan pengaturan kepemilikan bucket.

Saat menggunakan Replikasi Batch, Anda mungkin akan menjumpai salah satu kesalahan berikut:

- Status operasi batch gagal karena alasan: Laporan pekerjaan tidak dapat ditulis ke bucket laporan Anda.

Kesalahan ini terjadi jika peran IAM yang digunakan untuk pekerjaan Operasi Batch tidak dapat menempatkan laporan penyelesaian ke lokasi yang ditentukan saat Anda membuat pekerjaan. Untuk mengatasi kesalahan ini, periksa apakah peran IAM memiliki izin `PutObject` untuk bucket tempat Anda ingin menyimpan laporan penyelesaian Operasi Batch. Ini adalah praktik terbaik untuk mengirimkan laporan ke bucket yang berbeda dari bucket sumber.

- Operasi Batch selesai dengan kegagalan dan Total gagal bukan 0.

  Kesalahan ini terjadi jika terdapat masalah izin objek yang tidak memadai dengan pekerjaan Replikasi Batch yang sedang berjalan. Jika Anda menggunakan aturan replikasi untuk pekerjaan Replikasi Batch, pastikan bahwa peran IAM yang digunakan untuk replikasi memiliki izin yang tepat untuk mengakses objek baik dari bucket sumber atau tujuan. Anda juga dapat memeriksa Laporan penyelesaian Replikasi Batch untuk meninjau alasan kegagalan replikasi Amazon S3 tertentu.
- Pekerjaan batch berhasil berjalan tetapi jumlah objek yang diharapkan di bucket tujuan tidak sama.

  Kesalahan ini terjadi ketika ada ketidakcocokan antara objek yang tercantum dalam manifes yang disediakan dalam tugas Replikasi Batch dan filter yang Anda pilih saat membuat pekerjaan. Anda mungkin juga menerima pesan ini jika objek di bucket sumber Anda tidak cocok dengan aturan replikasi apa pun dan tidak disertakan dalam manifes yang dihasilkan.

Memecahkan masalah pencatatan akses server

Topik berikut dapat membantu Anda memecahkan masalah yang mungkin Anda alami saat menyiapkan pencatatan dengan Amazon S3.

Topik

- Pesan kesalahan umum saat mengatur pencatatan
- Memecahkan masalah kegagalan pengiriman

Pesan kesalahan umum saat mengatur pencatatan

Pesan galat umum berikut dapat muncul saat Anda mengaktifkan pencatatan melalui AWS Command Line Interface (AWS CLI) dan SDK AWS:

Kesalahan: Pencatatan lokasi Cross S3 tidak diizinkan

Jika bucket tujuan (juga dikenal sebagai bucket target) berada di Region yang berbeda dari bucket sumber, kesalahan Pencatatan lokasi Cross S3 tidak diizinkan akan terjadi. Untuk mengatasi kesalahan ini, pastikan bucket tujuan yang dikonfigurasi untuk menerima log akses berada dalam Wilayah AWS dan Akun AWS yang sama dengan bucket sumber.

Kesalahan: Pemilik bucket yang akan dicatat dan bucket target harus sama

Saat Anda mengaktifkan pencatatan akses server, kesalahan ini terjadi jika bucket tujuan yang ditentukan dimiliki akun lain. Untuk mengatasi kesalahan ini, pastikan bucket tujuan yang dikonfigurasi untuk menerima log akses berada dalam Akun AWS yang sama dengan bucket sumber.

> **Note**
>
> Kami menyarankan Anda memilih bucket tujuan yang berbeda dari bucket sumber. Ketika bucket sumber dan bucket tujuan sama, log tambahan dibuat untuk log yang ditulis ke bucket, yang dapat menambah tagihan penyimpanan Anda. Log tambahan tentang log ini juga dapat menyulitkan untuk menemukan log tertentu yang Anda cari. Untuk manajemen pencatatan yang lebih sederhana, kami sarankan untuk menyimpan log akses dalam bucket yang berbeda. Untuk informasi selengkapnya, lihat the section called “Bagaimana cara mengaktifkan log pengiriman?”.

Kesalahan: Bucket target untuk pencatatan tidak ada

Bucket tujuan harus ada sebelum menyetel konfigurasi. Kesalahan ini menunjukkan bahwa bucket tujuan tidak ada atau tidak dapat ditemukan. Pastikan bahwa nama bucket dieja dengan benar, lalu coba lagi.

Kesalahan: Pemberian target tidak diizinkan untuk bucket yang diberlakukan pemilik bucket

Kesalahan ini menunjukkan bahwa bucket tujuan menggunakan pengaturan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek S3. Pengaturan yang diberlakukan pemilik Bucket tidak mendukung pemberian tujuan (target). Untuk informasi selengkapnya, lihat Izin untuk pengiriman log.

Memecahkan masalah kegagalan pengiriman

Untuk menghindari masalah pencatatan akses server, pastikan Anda mengikuti praktik terbaik berikut ini:

- Grup pengiriman log S3 memiliki akses tulis ke bucket tujuan – Grup pengiriman log S3 mengirimkan log akses server ke bucket tujuan. Kebijakan bucket atau bucket daftar kontrol akses (ACL) dapat digunakan untuk memberikan akses tulis ke bucket tujuan. Namun, kami menyarankan Anda menggunakan kebijakan bucket, bukan ACL. Untuk informasi selengkapnya tentang cara memberikan akses tulis ke bucket tujuan, lihat Izin untuk pengiriman log.

> **Note**
>
> Jika bucket tujuan menggunakan setelan yang diberlakukan pemilik Bucket untuk Kepemilikan Objek, perhatikan hal-hal berikut:
>
> - ACL dinonaktifkan dan tidak lagi memengaruhi izin. Ini berarti Anda tidak dapat memperbarui ACL bucket untuk memberikan akses ke grup pengiriman log S3. Sebagai gantinya, untuk memberikan akses ke pengguna utama layanan pencatatan, Anda harus memperbarui kebijakan bucket untuk bucket tujuan.
> - Anda tidak dapat menyertakan pemberian tujuan dalam konfigurasi `PutBucketLogging` Anda.

- Kebijakan bucket untuk bucket tujuan memungkinkan akses ke log – Periksa kebijakan bucket dari bucket tujuan. Cari kebijakan bucket untuk setiap pernyataan yang berisi `"Effect": "Deny"`. Kemudian, verifikasi bahwa pernyataan `Deny` tersebut tidak mencegah log akses ditulis ke bucket.
- Kunci Objek S3 tidak diaktifkan di bucket tujuan – Periksa apakah bucket tujuan mengaktifkan Kunci Objek. Kunci Objek memblokir pengiriman log akses server. Anda harus memilih bucket tujuan yang tidak mengaktifkan Kunci Objek.
- Kunci terkelola Amazon S3 (SSE-S3) dipilih jika enkripsi default diaktifkan pada bucket tujuan – Anda dapat menggunakan enkripsi bucket default pada bucket tujuan hanya jika Anda menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3). Enkripsi sisi server dengan AWS Key Management Service (AWS KMS) kunci (SSE-KMS) tidak didukung untuk bucket tujuan pencatatan akses server. Untuk informasi selengkapnya tentang cara mengaktifkan enkripsi default, lihatMengonfigurasi enkripsi default.
- Bucket tujuan tidak mengaktifkan Pembayaran Pemohon – Menggunakan bucket Pembayaran Pemohon sebagai bucket tujuan untuk pencatatan akses server tidak didukung. Untuk mengizinkan pengiriman log akses server, nonaktifkan opsi Pembayaran Pemohon di bucket tujuan.
- Tinjau kebijakan kontrol layanan AWS Organizations Anda – Saat Anda menggunakan AWS Organizations, periksa kebijakan kontrol layanan untuk memastikan akses Amazon S3 diizinkan. Kebijakan kontrol layanan menentukan izin maksimum untuk akun yang terpengaruh. Cari kebijakan kontrol layanan untuk setiap pernyataan yang berisi `"Effect": "Deny"` dan verifikasi

bahwa pernyataan `Deny` tidak mencegah log akses apa pun ditulis ke bucket. Untuk informasi selengkapnya , lihat Kebijakan Kontrol Layanan (SCP) di Panduan Pengguna AWS Organizations.

- Berikan beberapa waktu agar perubahan konfigurasi pencatatan terbaru diterapkan – Mengaktifkan pencatatan akses server untuk pertama kalinya, atau mengubah bucket tujuan untuk log, memerlukan waktu untuk sepenuhnya diterapkan. Mungkin diperlukan waktu lebih dari satu jam agar semua permintaan dicatat dan dikirim dengan benar.

  Untuk memeriksa kegagalan pengiriman log, aktifkan metrik permintaan di Amazon CloudWatch. Jika log tidak dikirim dalam beberapa jam, cari metrik `4xxErrors`, yang dapat menunjukkan kegagalan pengiriman log. Untuk informasi lebih lanjut tentang mengaktifkan metrik permintaan, lihat the section called “Membuat konfigurasi metrik untuk semua objek”.

Pecahkan masalah Penentuan Versi

Topik berikut ini dapat membantu Anda memecahkan beberapa masalah umum Penentuan Versi Amazon S3.

Topik

- Saya ingin memulihkan objek yang secara tidak sengaja dihapus dalam bucket dengan Penentuan Versi diaktifkan
- Saya ingin menghapus objek berversi secara permanen
- Saya mengalami penurunan kinerja setelah mengaktifkan Penentuan Versi bucket

Saya ingin memulihkan objek yang secara tidak sengaja dihapus dalam bucket dengan Penentuan Versi diaktifkan

Secara umum, ketika versi objek dihapus dari bucket S3, tidak ada cara bagi Amazon S3 untuk memulihkannya. Namun, jika Anda sudah mengaktifkan Penentuan Versi S3 di bucket S3, permintaan `DELETE` yang tidak menentukan ID versi tidak dapat menghapus objek secara permanen. Sebagai gantinya, penanda hapus ditambahkan sebagai placeholder. Penanda hapus ini menjadi versi objek saat ini.

Untuk memverifikasi apakah objek yang dihapus telah dihapus secara permanen atau dihapus sementara (ditandai dengan penanda hapus), lakukan hal berikut:

1. Masuk ke AWS Management Console dan buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel navigasi kiri, pilih Bucket.
3. Di dalam daftar Bucket, pilih nama bucket yang berisi objek.
4. Pada daftar Objek, Aktifkan tombol Tampilkan versi di sebelah kanan bilah pencarian, lalu cari objek yang dihapus di bilah pencarian. Tombol beralih ini hanya tersedia jika Penentuan Versi sebelumnya telah diaktifkan di bucket.

   Anda juga dapat menggunakan Inventaris S3 untuk mencari objek yang dihapus.
5. Jika Anda tidak dapat menemukan objek setelah mengalihkan Tampilkan versi atau membuat laporan inventaris, dan Anda juga tidak dapat menemukan penanda hapus objek, penghapusan bersifat permanen dan objek tidak dapat dipulihkan.

Anda juga dapat memverifikasi status objek yang dihapus dengan menggunakan operasi `HeadObject` API dari AWS Command Line Interface (AWS CLI). Untuk menggunakannya, ganti perintah `head-object` dengan *`user input placeholders`* informasi Anda sendiri:

```
aws s3api head-object --bucket example-s3-bucket --key index.html
```

Jika Anda menjalankan perintah `head-object` pada objek berversi yang versinya saat ini adalah penanda hapus, Anda akan menerima kesalahan 404 Tidak Ditemukan. Sebagai contoh:

Terjadi kesalahan (404) saat memanggil HeadObject operasi: Tidak Ditemukan

Jika Anda menjalankan perintah `head-object` pada objek berversi dan memberikan ID versi objek, Amazon S3 mengambil metadata objek, mengonfirmasi bahwa objek tersebut masih ada dan tidak dihapus secara permanen.

```
aws s3api head-object --bucket example-s3-bucket --key index.html --
version-id versionID
```

```
{
"AcceptRanges": "bytes",
"ContentType": "text/html",
"LastModified": "Thu, 16 Apr 2015 18:19:14 GMT",
"ContentLength": 77,
"VersionId": "Zg5HyL7m.eZU9iM7AVlJkrqAiE.0UG4q",
"ETag": "\"30a6ec7e1a9ad79c203d05a589c8b400\"",
```

```
"Metadata": {}
}
```

Jika objek ditemukan dan versi terbaru adalah penanda hapus, versi objek sebelumnya masih akan ada. Karena penanda hapus adalah versi objek saat ini, Anda dapat memulihkan objek dengan menghapus penanda hapus.

Setelah Anda menghapus penanda hapus secara permanen, versi terbaru kedua dari objek akan menjadi versi objek saat ini, sehingga objek Anda akan kembali tersedia. Untuk gambaran visual tentang bagaimana objek dipulihkan, lihat Menghapus penanda hapus.

Untuk menghapus versi objek tertentu, Anda harus menjadi pemilik bucket. Untuk menghapus penanda hapus secara permanen, Anda harus menyertakan ID versinya dalam permintaan `DeleteObject`. Untuk menghapus penanda hapus, gunakan perintah berikut, lalu ganti *user input placeholders* dengan informasi Anda sendiri:

```
aws s3api delete-object --bucket example-s3-bucket --key index.html --
version-id versionID
```

Untuk informasi selengkapnya tentang perintah `delete-object`, lihat delete-object di Referensi Perintah AWS CLI . Untuk informasi selengkapnya tentang cara menghapus penanda hapus secara permanen, lihat. Mengelola penanda hapus

Saya ingin menghapus objek berversi secara permanen

Pada bucket dengan Penentuan Versi diaktifkan, permintaan `DELETE` tanpa ID versi tidak dapat menghapus objek secara permanen. Sebaliknya, permintaan semacam itu akan menyisipkan penanda hapus.

Untuk menghapus objek berversi secara permanen, Anda dapat memilih dari metode berikut:

- Buat aturan Siklus Hidup S3 untuk menghapus versi tidak terkini secara permanen. Untuk menghapus versi tidak terkini secara permanen, pilih Hapus versi objek tidak terkini secara permanen, lalu masukkan nomor di bawah Hari setelah objek menjadi tidak terkini. Anda dapat secara opsional menentukan jumlah versi yang lebih baru untuk dipertahankan dengan memasukkan nilai di bawah Jumlah versi yang lebih baru untuk dipertahankan. Untuk informasi selengkapnya tentang membuat aturan ini, lihat Menyetel konfigurasi Siklus Hidup S3.
- Hapus versi tertentu dengan menyertakan ID versi dalam permintaan `DELETE`. Untuk informasi selengkapnya, lihat Cara menghapus objek berversi secara permanen.

- Buat aturan siklus hidup untuk mengakhiri versi saat ini. Untuk mengakhiri versi objek saat ini, pilih Akhiri versi objek saat ini, lalu tambahkan angka di bawah Hari setelah pembuatan objek. Untuk informasi selengkapnya tentang membuat aturan siklus hidup ini, lihat Mengatur konfigurasi Siklus Hidup S3.
- Untuk menghapus semua objek berversi dan menghapus penanda secara permanen, buat dua aturan siklus hidup: satu untuk mengakhiri versi saat ini dan menghapus versi objek yang tidak terkini secara permanen, dan yang lainnya untuk menghapus penanda hapus objek yang kedaluwarsa.

Dalam bucket dengan Penentuan Versi yang diaktifkan, permintaan `DELETE` yang tidak menentukan ID versi hanya dapat menghapus objek dengan ID versi `NULL`. Jika objek diunggah saat Penentuan Versi diaktifkan, permintaan `DELETE`yang tidak menentukan ID versi akan membuat penanda hapus dari objek tersebut.

> **Note**
>
> Untuk bucket berkemampuan Kunci Objek S3, permintaan objek `DELETE` dengan ID versi objek yang dilindungi menyebabkan kesalahan 403 Akses Ditolak. Permintaan objek `DELETE` tanpa ID versi menambahkan penanda hapus sebagai versi terbaru dari objek dengan respons 200 OK. Objek yang dilindungi oleh Kunci Objek tidak dapat dihapus secara permanen sampai periode retensi dan penahanan hukumnya dihapus. Untuk informasi selengkapnya, lihat the section called “Cara kerja Kunci Objek S3”.

Saya mengalami penurunan kinerja setelah mengaktifkan Penentuan Versi bucket

Penurunan kinerja dapat terjadi pada bucket berkemampuan Penentuan Versi jika ada terlalu banyak penanda hapus atau objek berversi, dan jika praktik terbaik tidak diikuti.

Terlalu banyak penanda hapus

Setelah Anda mengaktifkan Penentuan Versi pada bucket, permintaan `DELETE` tanpa ID versi yang dibuat pada objek akan membuat penanda hapus dengan ID versi unik. Konfigurasi siklus hidup dengan aturan Akhiri versi objek saat ini menambahkan penanda hapus dengan ID versi unik ke setiap objek. Penanda hapus yang berlebihan dapat mengurangi kinerja dalam bucket.

Saat Penentuan Versi ditangguhkan pada bucket, Amazon S3 menandai ID versi sebagai `NULL` pada objek yang baru dibuat. Tindakan pengakhiran dalam bucket yang ditangguhkan Penentuan Versi menyebabkan Amazon S3 membuat penanda hapus dengan `NULL` sebagai ID versi. Dalam bucket yang ditangguhkan Penentuan Versi, penanda hapus `NULL` dibuat untuk permintaan penghapusan apa pun. Penanda hapus `NULL` ini juga disebut penanda hapus objek kedaluwarsa ketika semua versi objek dihapus dan hanya satu penanda hapus yang tersisa. Jika terlalu banyak penanda hapus `NULL` yang terakumulasi, akan terjadi penurunan kinerja dalam bucket.

Terlalu banyak objek berversi

Jika bucket berkemampuan Penentuan Versi berisi objek dengan jutaan versi, peningkatan kesalahan 503 Layanan Tidak Tersedia dapat terjadi. Jika Anda menyadari adanya peningkatan signifikan dalam jumlah HTTP 503 Layanan Tidak Tersedia yang diterima untuk permintaan objek `PUT` atau `DELETE` ke bucket dengan Penentuan Versi yang diaktifkan, Anda kemungkinan memiliki satu atau beberapa objek dalam bucket yang memiliki jutaan versi. Jika Anda memiliki objek dengan jutaan versi, Amazon S3 secara otomatis membatasi permintaan ke bucket tersebut. Permintaan pembatasan melindungi bucket Anda dari jumlah lalu lintas permintaan yang berlebihan, yang dapat berpotensi mengganggu permintaan lain yang dilakukan ke bucket yang sama.

Untuk menentukan objek mana yang memiliki jutaan versi, gunakan Inventaris S3. S3 Inventory menghasilkan laporan yang menyediakan daftar file datar dari objek dalam bucket. Untuk informasi selengkapnya, lihat Inventaris Amazon S3.

Untuk memverifikasi apakah ada banyak objek berversi dalam bucket, gunakan metrik Lensa Penyimpanan S3 untuk melihat Jumlah objek versi saat ini, Jumlah objek versi tidak terkini, dan Jumlah objek penanda hapus. Untuk informasi lebih lanjut tentang metrik Lensa Penyimpanan, silakan lihat Glosarium metrik Amazon S3 Storage Lens.

Tim Amazon S3 menganjurkan pelanggan untuk menyelidiki aplikasi yang berulang kali menimpa objek yang sama, yang berpotensi menciptakan jutaan versi untuk objek tersebut, sehingga memastikan apakah aplikasi bekerja sebagaimana mestinya. Misalnya, aplikasi menimpa objek yang sama setiap menit selama seminggu dapat membuat lebih dari sepuluh ribu versi. Kami merekomendasikan menyimpan kurang dari seratus ribu versi untuk setiap objek. Jika Anda memiliki kasus penggunaan yang membutuhkan jutaan versi untuk satu atau lebih objek, hubungi AWS Support tim untuk bantuan dalam menentukan solusi yang lebih baik.

Praktik terbaik

Untuk mencegah masalah penurunan performa terkait Penentuan Versi, kami sarankan Anda menerapkan praktik terbaik berikut:

- Aktifkan aturan siklus hidup untuk mengakhiri versi objek sebelumnya. Misalnya, Anda dapat membuat aturan siklus hidup untuk mengakhiri versi tidak terkini setelah 30 hari objek menjadi tidak terkini. Anda juga dapat mempertahankan beberapa versi tidak terkini jika Anda tidak ingin menghapus semuanya. Untuk informasi selengkapnya, lihat Mengatur konfigurasi Siklus Hidup S3.
- Aktifkan aturan siklus hidup untuk menghapus penanda penghapusan objek kedaluwarsa yang tidak memiliki objek data terkait di bucket. Untuk informasi selengkapnya, lihat Menghapus penanda hapus objek kedaluwarsa.

Untuk praktik terbaik pengoptimalan kinerja Amazon S3 tambahan, lihat Pola desain praktik terbaik.

Mendapatkan ID permintaan Amazon S3 untuk AWS Support

Setiap kali Anda menghubungi AWS Support karena mengalami kesalahan atau perilaku tak terduga di Amazon S3, Anda harus memberikan ID permintaan yang terkait dengan tindakan yang gagal. AWS Support menggunakan ID permintaan ini untuk membantu menyelesaikan masalah yang Anda alami.

ID Permintaan hadir berpasangan, dikembalikan dalam setiap respons yang diproses Amazon S3 (bahkan respons yang salah), dan dapat diakses melalui log verbose. Ada sejumlah metode umum untuk mendapatkan ID permintaan Anda, termasuk log akses S3 dan AWS CloudTrail peristiwa atau peristiwa data.

Setelah Anda memulihkan log ini, salin dan pertahankan kedua nilai tersebut, karena Anda akan membutuhkannya saat menghubungi AWS Support. Untuk informasi tentang menghubungi AWS Support, lihat Kontak AWS atau AWS Support Dokumentasi.

Menggunakan HTTP untuk memperoleh ID Permintaan

Anda dapat memperoleh ID permintaan Anda, `x-amz-request-id` dan `x-amz-id-2` dengan mencatat bit dari permintaan HTTP sebelum itu mencapai aplikasi target. Ada berbagai alat pihak ketiga yang dapat digunakan untuk memulihkan log verbose untuk permintaan HTTP. Pilih satu yang Anda percaya, lalu jalankan alat untuk mendengarkan di port yang dilalui lalu lintas Amazon S3 Anda, saat Anda mengirim permintaan HTTP Amazon S3 lainnya.

Untuk permintaan HTTP, sepasang ID permintaan akan terlihat seperti berikut:

```
x-amz-request-id: 79104EXAMPLEB723
x-amz-id-2: IOWQ4fDEXAMPLEQM+ey7N9WgVhSnQ6JEXAMPLEZb7hSQDASK+Jd1vEXAMPLEa3Km
```

> **Note**
>
> Permintaan HTTPS dienkripsi dan tersembunyi di sebagian besar tangkapan paket.

Menggunakan peramban web untuk memperoleh ID permintaan

Sebagian besar peramban web memiliki alat pengembang yang dapat Anda gunakan untuk melihat header permintaan.

Untuk permintaan berbasis peramban web yang memunculkan kesalahan, pasangan ID permintaan akan terlihat seperti contoh berikut.

```
<Error><Code>AccessDenied</Code><Message>Access Denied</Message>
<RequestId>79104EXAMPLEB723</RequestId><HostId>IOWQ4fDEXAMPLEQM
+ey7N9WgVhSnQ6JEXAMPLEZb7hSQDASK+Jd1vEXAMPLEa3Km</HostId></Error>
```

Untuk mendapatkan pasangan ID permintaan dari permintaan yang berhasil, gunakan alat pengembang peramban Anda untuk melihat header respons HTTP. Untuk informasi tentang alat developer untuk peramban tertentu, lihat Pemecahan masalah Amazon S3-Cara memulihkan permintaan S3 di AWS re:Post.

Menggunakan AWS SDK untuk mendapatkan ID permintaan

Bagian berikut mencakup informasi untuk mengonfigurasi pencatatan menggunakan AWS SDK. Meskipun Anda dapat mengaktifkan pencatatan verbose di setiap permintaan dan respons, kami tidak menyarankan untuk mengaktifkan pencatatan di sistem produksi, karena permintaan atau respons yang besar dapat memperlambat aplikasi secara signifikan.

Untuk permintaan AWS SDK, pasangan ID permintaan akan terlihat seperti contoh berikut.

```
Status Code: 403, AWS Service: Amazon S3, AWS Request ID: 79104EXAMPLEB723
AWS Error Code: AccessDenied  AWS Error Message: Access Denied
S3 Extended Request ID: IOWQ4fDEXAMPLEQM+ey7N9WgVhSnQ6JEXAMPLEZb7hSQDASK
+Jd1vEXAMPLEa3Km
```

Menggunakan SDK for Go untuk mendapatkan ID permintaan

Anda dapat mengonfigurasi logging dengan menggunakan SDK for Go. Untuk informasi selengkapnya, lihat Metadata respons di Panduan Pengembang SDK for Go V2.

Menggunakan SDK for PHP untuk memperoleh ID Permintaan

Anda dapat mengonfigurasi pencatatan menggunakan PHP. Untuk informasi lebih lanjut, lihat Bagaimana cara melihat data apa yang dikirim melalui kabel? di Panduan Pengembang AWS SDK for PHP .

Menggunakan SDK for Java untuk memperoleh ID permintaan

Anda dapat mengaktifkan pencatatan untuk permintaan atau respons tertentu untuk menangkap dan mengembalikan hanya header yang relevan. Untuk melakukannya, impor kelas `com.amazonaws.services.s3.S3ResponseMetadata`. Setelah itu, Anda dapat menyimpan permintaan dalam variabel sebelum melakukan permintaan aktual. Untuk mendapatkan permintaan atau respons yang dicatat, panggil `getCachedResponseMetadata(AmazonWebServiceRequest request).getRequestID()`.

Example

```
PutObjectRequest req = new PutObjectRequest(bucketName, key, createSampleFile());
s3.putObject(req);
S3ResponseMetadata md = s3.getCachedResponseMetadata(req);
System.out.println("Host ID: " + md.getHostId() + " RequestID: " + md.getRequestId());
```

Atau, Anda dapat menggunakan pencatatan log verbose untuk setiap permintaan dan respons Java. Untuk informasi selengkapnya, lihat Verbose Wire Logging di Panduan Pengembang AWS SDK for Java .

Menggunakan AWS SDK for .NET untuk mendapatkan ID permintaan

Anda dapat mengonfigurasi logging AWS SDK for .NET dengan menggunakan alat `System.Diagnostics` logging bawaan. Untuk informasi selengkapnya, lihat posting Logging with the AWS SDK for AWS .NET Developer Blog.

> **Note**
>
> Secara default, log yang dikembalikan hanya berisi informasi kesalahan. Untuk mendapatkan ID permintaan, file konfigurasi harus memiliki `AWSLogMetrics` (dan secara opsional, `AWSResponseLogging`) ditambahkan.

Menggunakan SDK untuk Python (Boto3) untuk memperoleh ID permintaan

Dengan itu AWS SDK for Python (Boto3), Anda dapat mencatat tanggapan tertentu. Anda dapat menggunakan fitur ini agar hanya menangkap header yang relevan. Kode berikut menunjukkan cara mencatat bagian dari respons terhadap file:

```
import logging
import boto3
logging.basicConfig(filename='logfile.txt', level=logging.INFO)
logger = logging.getLogger(__name__)
s3 = boto3.resource('s3')
response = s3.Bucket(bucket_name).Object(object_key).put()
logger.info("HTTPStatusCode: %s", response['ResponseMetadata']['HTTPStatusCode'])
logger.info("RequestId: %s", response['ResponseMetadata']['RequestId'])
logger.info("HostId: %s", response['ResponseMetadata']['HostId'])
logger.info("Date: %s", response['ResponseMetadata']['HTTPHeaders']['date'])
```

Anda juga dapat menangkap pengecualian dan mencatat informasi yang relevan ketika pengecualian diajukan. Untuk informasi selengkapnya, lihat Membedakan informasi berguna dari respons kesalahan di Referensi API SDK untuk Python AWS (Boto).

Selain itu, Anda dapat mengonfigurasi Boto3 agar mengeluarkan log debug verbose menggunakan kode berikut:

```
import boto3
boto3.set_stream_logger('', logging.DEBUG)
```

Untuk informasi selengkapnya, lihat set_stream_logger di Referensi API SDK untuk Python AWS (Boto).

Menggunakan SDK for Ruby untuk memperoleh ID permintaan

Anda bisa mendapatkan ID permintaan Anda menggunakan SDK for Ruby Versi 1, 2, atau 3.

- Menggunakan SDK for Ruby-Versi 1– Anda dapat mengaktifkan pencatatan log HTTP wire secara global dengan baris kode berikut.

  ```
  s3 = AWS::S3.new(:logger => Logger.new($stdout), :http_wire_trace => true)
  ```

- Menggunakan SDK for Ruby-Versi 2 atau Versi 3– Anda dapat mengaktifkan pencatatan log HTTP wire secara global dengan baris kode berikut.

```
s3 = Aws::S3::Client.new(:logger => Logger.new($stdout), :http_wire_trace => true)
```

Untuk tips mendapatkan informasi kawat dari AWS klien, lihat Tip debugging: Mendapatkan informasi jejak kawat dari klien.

Menggunakan AWS CLI untuk mendapatkan ID permintaan

Untuk mendapatkan ID permintaan Anda saat menggunakan AWS Command Line Interface (AWS CLI), tambahkan `--debug` ke perintah Anda.

Menggunakan Windows PowerShell untuk mendapatkan ID permintaan

Untuk informasi tentang memulihkan log dengan Windows PowerShell, lihat posting blog Response Logging in AWS Tools for Windows PowerShell .NET Development.

Menggunakan peristiwa AWS CloudTrail data untuk mendapatkan ID permintaan

Bucket Amazon S3 yang dikonfigurasi dengan peristiwa CloudTrail data untuk mencatat operasi API tingkat objek S3 memberikan informasi terperinci tentang tindakan yang diambil oleh pengguna, peran, atau layanan di AWS Amazon S3. Anda dapat mengidentifikasi ID permintaan S3 dengan menanyakan CloudTrail peristiwa dengan Athena.

Menggunakan pencatatan log akses server S3 untuk mendapatkan ID permintaan

Bucket Amazon S3 yang dikonfigurasi untuk pencatatan log akses server Amazon S3 memberikan catatan yang detail untuk setiap permintaan yang dibuat ke bucket. Anda dapat mengidentifikasi ID permintaan S3 dengan melakukan kueri log akses server menggunakan Athena.

Riwayat dokumen

- Versi API saat ini:2006-03-01

Tabel berikut menjelaskan perubahan penting dalam setiap rilis Referensi API Amazon Simple Storage Service dan Panduan Pengguna Amazon S3. Untuk notifikasi tentang pembaruan dokumentasi ini, Anda dapat berlangganan ke umpan RSS.

Perubahan	Deskripsi	Tanggal
Inventaris Amazon S3 mendukung kunci kondisi s3: InventoryAccessibleOptional Fields	Inventaris Amazon S3 mendukung kunci Inventory AccessibleOptionalFields kondisi s3: untuk mengontro l apakah pengguna dapat menyertakan bidang metadata opsional dalam laporan mereka. Untuk informasi selengkapnya, lihat Mengontro l pembuatan konfigurasi laporan Inventaris S3.	Februari 20, 2024
Dukungan IPv6 untuk S3 di Outposts	Anda sekarang dapat mengakses S3 di bucket Outposts menggunakan IPv6 melalui S3 pada titik akhir dual-stack Outposts. Dukungan IPv6 untuk S3 di Outposts memungkinkan Anda mengelola bucket S3 di Outposts dan mengontrol sumber daya pesawat melalui jaringan IPv6.	Januari 16, 2024
Kelas penyimpanan Amazon S3 zona tunggal berkinerja	Amazon S3 Express One Zone adalah kelas penyimpan	28 November 2023

tinggi baru–S3 Express One Zone	an Amazon S3 tunggal berkinerja tinggi yang dibuat khusus untuk menghadir kan akses data milidetik satu digit yang konsisten untuk aplikasi Anda yang paling sensitif terhadap latensi. Untuk informasi selengkapnya, lihat S3 Express One Zone.	
Mountpoint untuk Amazon S3 menambahkan dukungan untuk S3 Express One Zone	Anda sekarang dapat memasang bucket direktori S3 Express One Zone dengan Mountpoint.	28 November 2023
Versi skema invokasi Lambda	Operasi Batch Amazon S3 memperkenalkan versi skema invokasi Lambda baru untuk digunakan dengan tugas Operasi Batch yang bekerja pada bucket direktori. Untuk informasi selengkapnya, lihat Menggunakan Lambda dan operasi batch Amazon S3 dengan bucket direktori.	28 November 2023
Tindakan impor untuk bucket direktori	Amazon S3 memperken alkan tindakan impor. Impor adalah metode yang disederha nakan untuk membuat tugas Operasi Batch Amazon S3 guna menyalin objek dari bucket tujuan umum ke bucket direktori. Untuk informasi selengkapnya, lihat Mengimpor objek ke dalam bucket direktori.	28 November 2023

Mengelola akses S3 dengan Pemberian Akses S3	Amazon S3 Access Grants memungkinkan Anda mengelola izin data dalam skala besar untuk prinsipal AWS Identity and Access Management (IAM) selain identitas direktori dari direktori perusahaan seperti. Azure AD Anda sekarang dapat menerapkan izin S3 dengan hak akses paling rendah dan dengan mudah menskalak an izin tersebut berdasark an kebutuhan bisnis Anda. Untuk informasi selengkapnya, lihat Mengelola akses dengan Pemberian Akses S3.	26 November 2023
Mountpoint untuk Amazon S3 menambahkan fitur caching	Dengan Mountpoint, Anda sekarang dapat mengonfig urasi caching untuk data yang diakses berulang kali.	22 November 2023

Pembuatan manifes Operasi Batch Amazon S3 yang Ditingkatkan	Sekarang Anda dapat mengarahkan Operasi Batch Amazon S3 untuk menghasil kan manifes secara otomatis berdasarkan kriteria filter objek yang Anda tentukan saat membuat tugas. Opsi ini tersedia untuk pekerjaan replikasi batch yang Anda buat di konsol Amazon S3, atau untuk jenis pekerjaan apa pun yang Anda buat dengan menggunakan AWS CLI AWS , SDK, atau Amazon S3 REST API. Untuk informasi selengkapnya, lihat Membuat tugas Operasi Batch Amazon S3.	22 November 2023
Bucket Amazon S3 yang ada sekarang dapat menambahk an konfigurasi Kunci Objek	Anda kini dapat mengaktif kan Kunci Objek di bucket Amazon S3 yang ada. Anda dapat menetapkan periode retensi dan penahanan hukum untuk bucket baru atau yang sudah ada. Untuk informasi selengkapnya, lihat Menggunakan Kunci Objek.	20 November 2023
Lensa Penyimpanan S3 meminta metrik untuk awalan	Lensa Penyimpanan S3 memperkenalkan metrik permintaan untuk awalan dalam bucket Amazon S3. Untuk informasi selengkapnya, lihat Kategori metrik.	17 November 2023

Grup Lensa Penyimpanan Amazon S3	Lensa Penyimpanan S3 memperkenalkan grup Lensa Penyimpanan, filter yang ditentukan khusus untuk objek berdasarkan metadata objek. Untuk informasi selengkap nya, lihat Bekerja dengan grup Lensa Penyimpanan Amazon S3.	15 November 2023
Kebijakan IAM baru	S3 on Outposts memperken alkan `AWSServiceRoleForS3OnOutposts` , peran terkait layanan untuk membantu mengelola sumber daya jaringan untuk Anda. Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan untuk S3 on Outposts.	3 Oktober 2023
Amazon S3 menyediakan waktu `Last-Modified` untuk menghapus penanda hapus	Amazon S3 menyediakan waktu `Last-Modified` penanda hapus di header respons operasi S3 `Head` dan API `Get`. Untuk informasi selengkapnya, lihat Bekerja dengan penanda hapus.	27 September 2023
Pembaruan Amazon S3 ke AWS kebijakan terkelola	Amazon S3 menambahk an izin `s3:Describe*` ke `AmazonS3ReadOnlyAccess` . Untuk informasi selengkapnya, lihat kebijakan terkelola AWS untuk Amazon S3.	11 Agustus 2023

Peningkatan waktu mulai untuk permintaan pemulihan Standar yang dilakukan melalui Operasi Batch S3	Pengambilan Standar untuk permintaan pemulihan yang dilakukan melalui Operasi Batch S3 sekarang dapat dimulai dalam beberapa menit. Untuk informasi selengkapnya, lihat Opsi Pengambilan Arsip.	9 Agustus 2023
Menambahkan Mountpoint, klien dengan throughput tinggi untuk memasang bucket Amazon S3 sebagai sistem file lokal.	Dengan Mountpoint, aplikasi Anda dapat mengakses objek yang disimpan di Amazon S3 melalui operasi file, memberi aplikasi Anda akses ke penyimpanan elastis dan throughput Amazon S3 melalui antarmuka file.	9 Agustus 2023
Enkripsi sisi server dua lapis dengan kunci (DSSE-KMS) AWS Key Management Service	Enkripsi sisi server dua lapis dengan kunci AWS Key Management Service (AWS KMS) (DSSE-KMS) menerapkan dua lapisan enkripsi ke objek saat diunggah ke Amazon S3. Untuk informasi selengkapnya, lihat Menggunakan enkripsi sisi server dua lapis dengan kunci. AWS KMS	13 Juni 2023

Amazon S3 memungkinkan Blokir Akses Publik S3 dan menonaktifkan daftar kontrol akses (ACL) S3 untuk semua bucket baru.	Amazon S3 sekarang secara otomatis mengaktifkan S3 Block Public Access dan menonaktifkan daftar kontrol akses S3 (ACL) untuk semua bucket S3 baru di semua Wilayah. AWS Untuk informasi selengkapnya, lihat Memblokir akses publik ke penyimpanan Amazon S3 dan Mengontrol kepemilikan objek serta menonaktifkan ACL untuk bucket Anda.	27 April 2023
Metrik Operasi Replikasi S3 yang Gagal	Amazon S3 menambahkan Amazon CloudWatch metrik baru untuk memantau kegagalan Replikasi S3. Untuk informasi selengkapnya, lihat Memantau progres dengan metrik replikasi.	5 April 2023
DNS privat	AWS PrivateLink untuk Amazon S3 sekarang mendukung DNS Pribadi. Untuk informasi lebih lanjut, lihat DNS Privat.	14 Maret 2023
Dukungan titik akses lintas akun di konsol Amazon S3	Amazon S3 sekarang mendukung pembuatan titik akses lintas akun dengan konsol Amazon S3. Untuk informasi selengkapnya, lihat Membuat titik akses.	14 Maret 2023

Amazon S3 on Outposts mendukung Replikasi S3 di Outposts	Dengan Replikasi S3 lokal, Anda dapat secara otomatis mereplikasi objek ke satu bucket tujuan Outposts atau ke beberapa bucket tujuan. Bucket tujuan dapat berada di Outposts yang berbeda AWS Outposts atau dalam Outposts yang sama dengan bucket sumber. Untuk informasi selengkapnya, lihat Mereplikasi objek untuk S3 on Outposts.	14 Maret 2023
Alias Titik Akses Lambda Objek Amazon S3	Saat Anda membuat Titik Akses Lambda Objek, Amazon S3 secara otomatis menghasilkan alias unik untuk Titik Akses Lambda Objek Anda. Anda dapat menggunakan alias ini alih-alih nama bucket Amazon S3 atau Amazon Resource Name (ARN) Titik Akses Lamda Objek dalam permintaan untuk operasi bidang data titik akses. Untuk informasi selengkapnya, lihat Cara menggunakan alias bergaya bucket untuk Titik Akses Object Lambda.	14 Maret 2023

Dukungan lintas akun Titik Akses Multiwilayah Amazon S3	Amazon S3 sekarang mendukung pembuatan Titik Akses Multiwilayah lintas akun dengan konsol Amazon S3. Untuk informasi selengkap nya, lihat Membuat Titik Akses Multiwilayah.	14 Maret 2023
Titik akses lintas akun	Amazon S3 mendukung pembuatan titik akses lintas akun. Anda dapat membuat titik akses lintas akun dengan menggunakan AWS Command Line Interface (AWS CLI) atau operasi `CreateAccessPoint` API REST. Untuk informasi selengkapnya, lihat Membuat titik akses.	30 November 2022
Amazon S3 mendukung kontrol failover untuk Titik Akses Multiwilayah Amazon S3	Amazon S3 memperken alkan kontrol failover untuk Titik Akses Multiwilayah. Kontrol ini memungkinkan Anda mengalihkan lalu lintas permintaan akses data S3 yang dirutekan melalui Titik Akses Multiwilayah Amazon S3 ke Wilayah AWS alternati f dalam beberapa menit untuk menguji dan membangun aplikasi yang sangat tersedia. Untuk informasi selengkapnya, lihat kontrol failover Titik Akses Multiwilayah Amazon S3.	28 November 2022

Lensa Penyimpanan Amazon S3 meningkatkan visibilitas seluruh organisasi dengan 34 metrik baru	Lensa Penyimpanan S3 memperkenalkan 34 metrik tambahan untuk mengungkap peluang pengoptimalan biaya yang lebih dalam, mengident ifikasi praktik terbaik perlindun gan data, dan meningkatkan kinerja alur kerja aplikasi. Untuk informasi selengkapnya, lihat metrik Lensa Penyimpan an S3.	17 November 2022
Amazon S3 mendukung tingkat permintaan pemulihan yang lebih tinggi untuk S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive	Amazon S3 mendukung permintaan pemulihan dengan kecepatan hingga 1.000 transaksi per detik, per Akun AWS untuk kelas penyimpanan S3 Glacier Flexible Retrieval dan S3 Glacier Deep Archive.	15 November 2022

Amazon S3 on Outposts mendukung tindakan dan filter Siklus Hidup S3	S3 on Outposts mendukung aturan Siklus Hidup S3 tambahan untuk mengoptim alkan manajemen kapasitas. Anda dapat mengakhiri objek seiring bertambahnya usia atau diganti dengan versi yang lebih baru. Anda dapat membuat aturan siklus hidup untuk seluruh bucket atau subset objek dalam bucket dengan melakukan filter dengan awalan, tag objek, atau ukuran objek. Untuk informasi selengkapnya, lihat Membuat dan mengelola konfigurasi siklus hidup.	2 November 2022
Dukungan Replikasi S3 untuk objek SSE-C	Anda dapat mereplikasi objek yang dibuat menggunak an enkripsi di sisi server dengan kunci yang disediaka n pelanggan. Untuk informasi selengkapnya tentang mereplikasi objek terenkripsi, lihat Mereplikasi objek yang dibuat dengan enkripsi di sisi server (SSE-C, SSE-S3, SSE-KMS).	24 Oktober 2022

Amazon S3 on Outposts mendukung alias titik akses	Dengan S3 di Outposts, Anda harus menggunakan titik akses untuk mengakses objek apa pun dalam bucket Outposts. Setiap kali Anda membuat titik akses untuk bucket, S3 di Outposts secara otomatis menghasil kan alias titik akses. Anda dapat menggunakan alias titik akses ini alih-alih ARN titik akses untuk operasi bidang data apa pun. Untuk informasi selengkapnya, lihat Menggunakan alias bergaya bucket untuk titik akses bucket S3 on Outposts.	21 Oktober 2022
S3 Object Lambda mendukung operasi `HeadObject` , `ListObjects` , dan `ListObjectsV2`	Anda dapat menggunakan kode kustom untuk memodifik asi data yang dikembalikan oleh permintaan S3 `GET`, `LIST`, atau `HEAD` standar untuk melakukan filter baris, mengubah ukuran citra secara dinamis, menyamarkan data rahasia, dan banyak lagi. Untuk informasi selengkapnya, lihat Mengubah objek dengan S3 Object Lambda.	4 Oktober 2022

Amazon S3 on Outposts mendukung Penentuan Versi S3	Saat diaktifkan, S3 Penentuan Versi menyimpan beberapa salinan objek yang berbeda dalam bucket yang sama. Anda dapat menggunakan penentuan versi S3 untuk menyimpan, mengambil, dan memulihkan setiap versi dari setiap objek yang disimpan dalam bucket Outposts Anda. Penentuan Versi S3 membantu Anda memulihkan dari tindakan pengguna yang tidak diinginkan dan kegagalan aplikasi. Untuk informasi selengkapnya, lihat Mengelola Penentuan Versi S3 untuk bucket S3 on Outposts.	21 September 2022
AWS Backup untuk Amazon S3	AWS Backup adalah layanan berbasis kebijakan yang dikelola sepenuhnya yang dapat Anda gunakan untuk menentukan kebijakan pencadangan pusat untuk melindungi data Amazon S3 Anda. Untuk informasi selengkapnya, lihat Menggunakan AWS Backup untuk Amazon S3.	18 Februari 2022

Gunakan Replikasi Batch S3 untuk mereplikasi objek yang ada	Dengan Replikasi Batch S3, Anda dapat mereplikasi objek yang ada sebelum konfigurasi replikasi dilakukan. Mereplikasi objek yang ada dilakukan melalui penggunaan tugas Operasi Batch. Replikasi Batch S3 berbeda dengan replikasi langsung, yang secara terus menerus dan otomatis menyalin objek baru di seluruh bucket Amazon S3. Untuk informasi selengkapnya, lihat Mereplikasi objek yang ada dengan Replikasi Batch S3.	8 Februari 2022
Mengganti nama S3 Glacier Instant Retrieval	Kelas penyimpanan Glacier telah diubah namanya menjadi S3 Glacier Flexible Retrieval. Perubahan ini tidak memengaruhi API.	30 November 2021

Pengaturan Kepemilikan Objek S3 untuk menonaktifkan ACL	Anda dapat menerapkan pengaturan yang diberlakukan pemilik bucket untuk Kepemilikan Objek guna menonaktifkan ACL untuk bucket dan objek di dalamnya dan mengambil kepemilikan setiap objek di bucket Anda. Pengaturan yang diberlakukan pemilik bucket menyederhanakan manajemen akses untuk data yang disimpan di Amazon S3. Untuk informasi selengkapnya, lihat Mengontrol kepemilikan objek dan menonaktifkan ACL untuk bucket Anda.	30 November 2021
Kelas penyimpanan S3 Intelligent-Tiering	S3 Intelligent-Tiering Archive Instant Access adalah kelas penyimpanan tambahan dalam S3 Intelligent-Tiering. Untuk informasi selengkapnya, lihat Cara kerja S3 Intelligent-Tiering.	30 November 2021
Kelas penyimpanan S3 Glacier Instant Retrieval	Anda sekarang dapat menempatkan objek dalam kelas penyimpanan S3 Glacier Instant Retrieval. Untuk informasi selengkapnya tentang kelas penyimpanan ini, lihat Menggunakan kelas penyimpanan Amazon S3.	30 November 2021

AWS Backup untuk Pratinjau Amazon S3	AWS Backup adalah layanan berbasis kebijakan yang dikelola sepenuhnya yang dapat Anda gunakan untuk menentukan kebijakan pencadangan pusat untuk melindungi data Amazon S3 Anda. Untuk informasi selengkapnya lihat, Menggunakan AWS Backup untuk Amazon S3.	30 November 2021
AWS Identity and Access Management Access Analyzer untuk Amazon S3	Penganalisis Akses IAM menjalankan pemeriksaan kebijakan untuk memvalida si kebijakan Anda terhadap tata bahasa kebijakan IAM dan praktik terbaik. Untuk mempelajari validasi kebijakan menggunakan IAM Access Analyzer lebih lanjut, lihat Validasi kebijakan IAM Access Analyzer di Panduan Pengguna IAM.	30 November 2021
Tipe peristiwa baru	Tipe peristiwa baru yang ditambahkan ke Notifikasi Peristiwa Amazon S3, lihat Notifikasi Peristiwa Amazon S3.	29 November 2021
Aktifkan Amazon EventBridge di ember	Anda dapat mengaktifkan bucket EventBridge Amazon S3 untuk mengirim acara ke Amazon EventBridge, lihat Menggunakan. EventBridge	29 November 2021

Filter Siklus Hidup S3 baru	Anda dapat membuat aturan siklus hidup berdasarkan ukuran objek atau menentuka n berapa banyak versi objek lama yang akan disimpan. Untuk informasi selengkapnya, lihat Contoh konfigurasi Siklus Hidup S3.	23 November 2021
Publikasikan metrik Lensa Penyimpanan Amazon S3 ke Amazon CloudWatch	Anda dapat memublikasikan metrik penggunaan dan aktivitas Lensa Penyimpanan S3 CloudWatch ke Amazon untuk membuat tampilan terpadu kesehatan operasion al Anda di dasbor. CloudWatc h Anda juga dapat menggunak an CloudWatch fitur, seperti alarm dan tindakan yang dipicu, matematika metrik, dan deteksi anomali, untuk memantau dan mengambil tindakan pada metrik Lensa Penyimpanan S3. Selain itu, CloudWatch API memungkin kan aplikasi, termasuk penyedia pihak ketiga, untuk mengakses metrik Lensa Penyimpanan S3 Anda. Untuk informasi selengkapnya, lihat metrik Lensa Penyimpanan Monitor S3 di. CloudWatch	22 November 2021

Titik Akses Multiwilayah	Anda dapat menggunakan Titik Akses Multiwilayah untuk membuat titik akhir global yang dapat digunakan aplikasi untuk memenuhi permintaan dari bucket Amazon S3 yang terletak di beberapa Wilayah AWS. Anda dapat menggunakan Titik Akses Multiwilayah ini untuk merutekan data ke bucket dengan latensi terendah. Untuk informasi selengkapnya tentang Titik Akses Multiwilayah dan cara menggunakannya, lihat Titik Akses Multiwilayah di Amazon S3.	2 September 2021
Amazon S3 on Outposts menambahkan akses lokal langsung untuk aplikasi	Jalankan aplikasi Anda di luar AWS Outposts virtual private cloud (VPC) dan akses data S3 Anda di Outposts. Anda juga dapat mengakses objek S3 on Outposts langsung dari jaringan on-premise Anda. Untuk informasi selengkap nya tentang mengonfigurasi S3 pada titik akhir Outposts menggunakan alamat IP (CoIP) milik pelanggan dan mengakses objek Anda dengan membuat gateway lokal dari jaringan on-premis e, lihat Mengakses Amazon S3 on Outposts menggunakan titik akses khusus VPC.	29 Juli 2021

Alias titik akses Amazon S3	Saat Anda membuat titik akses, Amazon S3 secara otomatis membuat alias yang dapat Anda gunakan alih-alih nama bucket untuk akses data. Anda dapat menggunakan alias titik akses ini alih-alih Amazon Resource Name (ARN) untuk operasi bidang data titik akses apa pun. Untuk informasi selengkapnya, lihat Menggunakan alias bergaya bucket untuk titik akses Anda.	26 Juli 2021
Inventaris Amazon S3 dan Operasi Batch S3 mendukung status Kunci Bucket S3	Inventaris Amazon S3 dan Operasi Batch mendukung pengidentifikasian dan penyalinan objek yang ada dengan Kunci Bucket S3. Kunci Bucket S3 mempercepat pengurangan biaya enkripsi di sisi server untuk objek yang sudah ada. Untuk informasi selengkapnya, lihat Inventaris Amazon S3 dan Objek Penyalinan Operasi Batch.	3 Juni 2021

Snapshot akun metrik Lensa Penyimpanan Amazon S3	Snapshot akun Lensa Penyimpanan S3 menampilkan total penyimpanan, jumlah objek, dan ukuran objek rata-rata Anda di halaman beranda konsol S3 (Bucket) dengan meringkas metrik dari dasbor default Anda. Untuk informasi selengkapnya, lihat Snapshot akun metrik Lensa Penyimpanan S3.	5 Mei 2021
Peningkatan dukungan titik akhir Amazon S3 on Outposts	S3 on Outposts sekarang mendukung hingga 100 titik akhir per Outpost. Untuk informasi selengkapnya, lihat Pembatasan jaringan S3 on Outposts.	29 April 2021
Amazon S3 tentang pemberitahuan acara Outposts di Acara Amazon CloudWatch	Anda dapat menggunakan CloudWatch Events untuk membuat aturan untuk menangkap S3 pada peristiwa Outposts API dan mendapatkan pemberitahuan melalui semua target yang didukung. CloudWatch Untuk informasi selengkapnya, lihat Menerima pemberitahuan CloudWatch acara S3 di Outposts menggunakan Acara.	19 April 2021

S3 Object Lambda	Dengan S3 Object Lambda, Anda dapat menambahkan kode Anda sendiri ke permintaan `GET` Amazon S3 untuk memodifikasi dan memproses data saat dikembalikan ke aplikasi. Anda dapat menggunakan kode khusus untuk memodifikasi data yang dikembalikan oleh permintaan `GET` S3 standar untuk melakukan filter baris, mengubah ukuran citra secara dinamis, menyamarkan data rahasia, dan banyak lagi. Untuk informasi selengkapnya, lihat Mengubah objek.	18 Maret 2021
AWS PrivateLink	Dengan AWS PrivateLink Amazon S3, Anda dapat terhubung langsung ke S3 dengan menggunakan titik akhir antarmuka di cloud pribadi virtual (VPC) Anda alih-alih terhubung melalui internet. Titik akhir antarmuka dapat diakses langsung dari aplikasi yang ada di on-premise atau di Wilayah AWS yang berbeda. Untuk informasi selengkapnya lihat, AWS PrivateLink untuk Amazon S3.	2 Februari 2021

Mengelola Amazon S3 pada kapasitas Outposts dengan AWS CloudTrail	S3 pada acara manajemen Outposts tersedia CloudTrai l melalui log. Untuk informasi selengkapnya, lihat Mengelola S3 pada kapasitas Outposts dengan. CloudTrail	21 Desember 2020
Konsistensi kuat	Amazon S3 memberikan read-after-write konsisten si yang kuat untuk PUT dan DELETE permintaan objek di bucket S3 Anda secara keseluruhan. Wilayah AWS Selain itu, operasi baca di Amazon S3 Select, daftar kontrol akses Amazon S3, Amazon S3 Object Tag, dan metadata objek (misalnya, objek HEAD) sangat konsisten. Untuk informasi selengkapnya, lihat Model konsistensi data Amazon S3.	1 Desember 2020

Sinkronisasi modifikasi replika Amazon S3	Sinkronisasi modifikasi replika Amazon S3 menjaga agar metadata objek seperti tag, ACL, dan pengaturan Kunci Objek tetap sinkron antara objek sumber dan replika. Saat fitur ini diaktifkan, Amazon S3 mereplikasi perubahan metadata yang dibuat ke objek sumber atau salinan replika. Untuk informasi selengkapnya, lihat Mereplikasi perubahan metadata dengan sinkronisasi modifikasi replika.	1 Desember 2020
Kunci Bucket Amazon S3	Kunci Bucket Amazon S3 mengurangi biaya enkripsi di sisi server Amazon S3 dengan (SSE-KMS) AWS Key Management Service . Kunci level bucket baru untuk enkripsi di sisi server ini dapat mengurangi biaya permintaan AWS KMS hingga 99 persen dengan mengurangi lalu lintas permintaan dari Amazon S3 ke AWS KMS. Untuk informasi selengkapnya, lihat Mengurangi biaya SSE-KMS menggunakan Kunci Bucket S3.	1 Desember 2020

Lensa Penyimpanan Amazon S3	Lensa Penyimpanan S3 menggabungkan metrik Anda dan menampilkan informasi di bagian Snapshot akun di halaman Bucket konsol Amazon S3. Lensa Penyimpan an S3 juga menyediak an dasbor interaktif yang dapat Anda gunakan untuk memvisualisasikan wawasan dan tren, menandai outlier, serta menerima rekomendasi untuk mengoptimalkan biaya penyimpanan dan menerapka n praktik terbaik perlindun gan data. Dasbor Anda memiliki opsi perincian untuk menghasilkan dan memvisual isasikan wawasan di tingkat organisasi, akun, Wilayah AWS, kelas penyimpanan, bucket, prefiks, atau grup Lensa Penyimpanan. Anda juga dapat mengirimkan ekspor metrik harian dalam format CSV atau Parquet ke bucket S3. Untuk informasi selengkapnya, lihat Menilai aktivitas penyimpanan dan penggunaan Anda dengan Lensa Penyimpanan S3.	18 November 2020

Menelusuri permintaan S3 menggunakan AWS X-Ray	Amazon S3 terintegrasi dengan X-Ray untuk menyebarkan konteks pelacakan dan memberi Anda satu rantai permintaan dengan simpul hulu dan hilir. Untuk informasi selengkapnya, lihat Melacak permintaan menggunakan X-Ray.	16 November 2020
Metrik Replikasi S3	Metrik Replikasi S3 memberikan metrik terperinci untuk aturan replikasi dalam konfigurasi replikasi Anda. Untuk informasi selengkapnya, lihat Metrik replikasi dan pemberitahuan peristiwa Amazon S3.	9 November 2020
S3 Intelligent-Tiering Archive Access dan Deep Archive Access	S3 Intelligent-Tiering Archive Access dan Deep Archive Access adalah tingkat penyimpanan tambahan dalam S3 Intelligent-Tiering. Untuk informasi selengkapnya, lihat Kelas penyimpanan untuk secara otomatis mengoptimalkan objek yang sering dan jarang diakses .	9 November 2020

Replikasi penanda hapus	Dengan replikasi penanda hapus, Anda dapat memastikan bahwa penanda hapus disalin ke bucket tujuan untuk aturan replikasi Anda. Untuk informasi selengkapnya, lihat Menggunakan replikasi penanda hapus.	9 November 2020
Kepemilikan Objek S3	Kepemilikan Objek adalah pengaturan bucket S3 yang dapat Anda gunakan untuk mengendalikan kepemilikan objek baru yang diunggah ke bucket Anda. Untuk informasi selegnkapnya, lihat Menggunakan Kepemilikan Objek S3.	2 Oktober 2020
Amazon S3 di Outposts	Dengan Amazon S3 di Outposts, Anda dapat membuat bucket S3 pada AWS Outposts sumber daya Anda dan dengan mudah menyimpan serta mengambil objek lokal untuk aplikasi yang memerlukan akses data lokal, pemrosesan data lokal, dan residensi data. Anda dapat menggunakan S3 di Outposts melalui AWS Managemen t Console,, SDK AWS CLI AWS , atau REST API. Untuk informasi selengkapnya, lihat Penggunaan Amazon S3 on Outposts.	30 September 2020

Kondisi pemilik bucket	Anda dapat menggunakan kondisi pemilik bucket Amazon S3 untuk memastikan bahwa bucket yang Anda gunakan dalam operasi S3 milik yang Anda harapkan. Akun AWS Untuk informasi selengkapnya, lihat Kondisi pemilik bucket.	11 September 2020
Dukungan Operasi Batch S3 untuk Retensi Kunci Objek	Sekarang, Anda dapat menggunakan Operasi Batch dengan Kunci Objek S3 untuk menerapkan pengaturan penyimpanan pada banyak objek Amazon S3 sekaligus . Untuk informasi selengkap nya, lihat Menetapkan tanggal Retensi Kunci Objek S3 dengan Operasi Batch S3.	4 Mei 2020
Dukungan Operasi Batch S3 untuk Penahanan Hukum Kunci Objek	Sekarang Anda dapat menggunakan Operasi Batch dengan Kunci Objek S3 untuk menambahkan penahanan hukum ke banyak objek Amazon S3 sekaligus. Untuk informasi selengkapnya, lihat Menggunakan Operasi Batch S3 untuk mengatur Penahanan Hukum Kunci Objek S3.	4 Mei 2020

Tag tugas untuk Operasi Batch S3	Anda dapat menambahkan tag ke tugas Operasi Batch S3 Anda untuk mengontrol dan memberi label tugas tersebut. Untuk informasi selengkapnya, lihat Tag untuk tugas Operasi Batch S3.	16 Maret 2020
Titik akses Amazon S3	Titik akses Amazon S3 menyederhanakan pengelola an akses data dalam skala besar untuk set data bersama di S3. Titik akses diberi nama titik akhir jaringan yang melekat pada bucket yang dapat Anda gunakan untuk melakukan operasi objek S3. Untuk informasi selengkapnya, lihat Mengelola akses data dengan titik akses Amazon S3.	2 Desember 2019
Access Analyzer untuk Amazon S3	Access Analyzer untuk Amazon S3 memberi tahu Anda tentang bucket S3 yang dikonfigurasi untuk memungkinkan akses ke siapa pun di internet atau Akun AWS lainnya, termasuk akun di luar organisasi Anda. Untuk informasi selengkapnya, lihat Menggunakan Access Analyzer untuk Amazon S3.	2 Desember 2019

Kontrol Waktu Replikasi S3 (S3 RTC)	Kontrol Waktu Replikasi S3 (S3 RTC) mereplikasi sebagian besar objek yang Anda unggah ke Amazon S3 dalam hitungan detik, dan 99,99 persen dari objek tersebut dalam waktu 15 menit. Untuk informasi selengkapnya, lihat Mereplikasi Objek menggunakan Kontrol Waktu Replikasi S3 (S3 RTC).	20 November 2019
Replikasi Wilayah yang Sama	Replikasi Wilayah yang Sama (SRR) digunakan untuk menyalin objek di seluruh bucket Amazon S3 dalam Wilayah AWS yang sama. Untuk informasi tentang Replikasi Lintas-Wilayah (CRR) dan Replikasi Wilayah yang Sama, lihat Replikasi.	18 September 2019
Dukungan Replikasi Lintas-Wilayah untuk Kunci Objek S3	Replikasi Lintas-Wilayah sekarang mendukung Kunci Objek. Untuk informasi selengkapnya, lihat Apa yang Direplikasi Amazon S3?.	28 Mei 2019

Operasi Batch S3	Dengan menggunakan Operasi Batch S3, Anda dapat melakukan Operasi Batch berskala besar pada objek Amazon S3. Operasi Batch S3 dapat menjalankan satu operasi pada daftar objek yang Anda tentukan. Satu tugas dapat melakukan operasi tertentu pada miliaran objek yang berisi data sebesar exabyte. Untuk informasi selengkapnya, lihat Melakukan Operasi Batch S3.	30 April 2019
Wilayah Asia Pasifik (Hong Kong)	Amazon S3 kini tersedia di Wilayah Asia Pasifik (Hong Kong). Untuk informasi selengkapnya tentang Wilayah dan titik akhir Amazon S3, lihat Wilayah dan titik akhir di Referensi Umum AWS.	24 April 2019
Menambahkan bidang baru ke log akses server	Amazon S3 menambahkan bidang baru berikut ke log akses server: versi Keamanan Lapisan Pengangkutan (TLS). Untuk informasi selengkapnya, lihat Format log akses server.	28 Maret 2019

Kelas penyimpanan arsip baru	Amazon S3 kini menawarkan kelas penyimpanan arsip baru, S3 Glacier Deep Archive (`DEEP_ARCHIVE`), untuk menyimpan objek yang jarang diakses. Untuk informasi selengkapnya, lihat Kelas Penyimpanan.	27 Maret 2019
Menambahkan bidang baru ke log akses server	Amazon S3 menambahkan bidang baru berikut ke log akses server: Host Id, Signature Version, Cipher Suite, Authentication Type, dan Host Header. Untuk informasi selengkapnya, lihat Format log akses server.	5 Maret 2019
Dukungan untuk berkas Inventaris Amazon S3 yang diformat oleh Parquet	Amazon S3 sekarang mendukung format Apache Parquet (Parquet) selain format file kolom baris yang dioptimalkan (ORC) Apache dan nilai yang dipisahkan koma (CSV) untuk file output persediaan. Untuk informasi selengkapnya, lihat Inventaris.	4 Desember 2018
Kunci Objek S3	Amazon S3 kini menawarkan fungsionalitas Kunci Objek yang menyediakan perlindungan Tulis Sekali Baca Banyak (WORM) untuk objek Amazon S3. Untuk informasi selengkapnya, lihat Mengunci Objek.	26 November 2018

Peningkatan kecepatan pemulihan	Dengan peningkatan kecepatan pemulihan Amazon S3, Anda dapat mengubah kecepatan pemulihan dari kelas penyimpanan S3 Glacier Flexible Retrieval menjadi lebih cepat saat pemulihan sedang berlangsung. Untuk informasi selengkapnya, lihat Memulihkan Objek yang Diarsipkan.	26 November 2018
Notifkasi Peristiwa Pemulihan	Notifikasi Peristiwa Amazon S3 kini mendukung peristiwa inisiasi dan penyelesaian saat memulihkan objek dari kelas penyimpanan S3 Glacier Flexible Retrieval. Untuk informasi selengkapnya, lihat Notifikasi Peristiwa.	26 November 2018

PUT langsung ke kelas penyimpanan S3 Glacier Flexible Retrieval	Operasi PUT Amazon S3 kini mendukung penentuan S3 Glacier Flexible Retrieval sebagai kelas penyimpan an saat membuat objek. Sebelumnya, Anda harus melakukan transisi objek ke kelas penyimpanan S3 Glacier Flexible Retrieval dari kelas penyimpanan Amazon S3 lainnya. Selain itu, saat menggunakan Replikasi Lintas-Wilayah (CRR) S3, Anda sekarang dapat menentukan S3 Glacier Flexible Retrieval sebagai kelas penyimpanan untuk objek yang direplikasi. Untuk informasi selengkapnya tentang kelas penyimpanan S3 Glacier Flexible Retrieval , lihat Kelas Penyimpanan. Untuk informasi selengkap nya tentang menentukan kelas penyimpanan untuk objek yang direplikasi, Ikhtisar Konfigurasi Replikasi. Untuk informasi selengkapnya tentang PUT langsung ke perubahan API REST S3 Glacier Flexible Retrieval , lihat Riwayat Dokumen: PUT langsung ke S3 Glacier Flexible Retrieval.	26 November 2018

Kelas penyimpanan baru	Amazon S3 kini menawarkan kelas penyimpanan baru bernama S3 Intelligent-Tiering (`INTELLIGENT_TIERING`) yang dirancang untuk data jangka panjang dengan pola akses yang berubah atau tidak diketahui. Untuk informasi selengkapnya, lihat Kelas Penyimpanan.	26 November 2018
Blokir Akses Publik Amazon S3	Amazon S3 kini menyertakan kemampuan untuk memblokir akses publik ke bucket dan objek dengan basis per bucket atau per akun. Untuk informasi selengkapnya, lihat Menggunakan Blokir Akses Publik Amazon S3.	15 November 2018
Peningkatan pemfilteran dalam aturan Replikasi Lintas-Wilayah (CRR)	Dalam konfigurasi aturan CRR, Anda dapat menentukan filter objek untuk memilih subset objek yang akan menerapkan aturan tersebut. Sebelumnya, Anda hanya dapat memfilter pada awalan kunci objek. Dalam rilis ini, Anda dapat memfilter awalan kunci objek, satu atau beberapa tag objek, atau keduanya. Untuk informasi selengkapnya, lihat Pengaturan CRR: Ikhtisar Konfigurasi Replikasi.	19 September 2018

Fitur baru Amazon S3 Select	Amazon S3 Select sekarang mendukung input Apache Parquet, kueri pada objek JSON bersarang, dan dua metrik pemantauan Amazon baru (dan). CloudWatch `SelectScannedBytes` `SelectReturnedBytes`	5 September 2018
Pembaruan kini tersedia melalui RSS	Anda sekarang dapat berlangganan umpan RSS untuk menerima pemberita huan tentang pembaruan pada Panduan Pengguna Amazon S3.	19 Juni 2018

Pembaruan sebelumnya

Tabel berikut menjelaskan perubahan penting dalam setiap rilis Panduan Pengguna Amazon S3 sebelum 19 Juni 2018.

Perubahan	Deskripsi	Tanggal
Pembaruan contoh kode	Contoh kode diperbarui: • C#—Memperbarui semua contoh untuk menggunak an pola asinkron berbasis tugas. Untuk informasi selengkapnya, lihat Amazon Web Services Asynchron ous API untuk.NET di Panduan Pengembang. AWS SDK for .NET Contoh kode sekarang sesuai dengan versi 3 dari AWS SDK for .NET. • Java—Memperbarui semua contoh untuk menggunak an model pembuat klien. Untuk informasi selengkap nya tentang model pembuat klien, lihat Membuat Klien Layanan.	30 April 2018

Perubahan	Deskripsi	Tanggal
	• PHP—Memperbarui semua contoh untuk menggunakan AWS SDK for PHP 3.0. Untuk informasi lebih lanjut tentang AWS SDK for PHP 3.0, lihat AWS SDK for PHP. • Ruby—Kode contoh yang diperbarui sehingga contoh bekerja dengan versi 3. AWS SDK for Ruby	

Amazon S3 sekarang melaporkan S3 Glacier Flexible Retrieval dan `ONEZONE_IA` kelas penyimpanan ke metrik penyimpanan Amazon Logs CloudWatch	Selain melaporkan byte aktual, metrik penyimpanan ini mencakup byte overhead per objek untuk kelas penyimpanan yang berlaku (`ONEZONE_IA` , `STANDARD_IA` , dan S3 Glacier Flexible Retrieval):
	• Untuk objek kelas penyimpanan `ONEZONE_IA` dan `STANDARD_IA` , Amazon S3 melaporkan objek lebih kecil dari 128 KB sebagai 128 KB. Untuk informasi selengkapnya, lihat Menggunakan kelas penyimpanan Amazon S3.
	• Untuk objek kelas penyimpanan S3 Glacier Flexible Retrieval, metrik penyimpanan melaporkan overhead berikut:
	  • Overhead 32 KB per objek, dikenakan harga kelas penyimpanan S3 Glacier Flexible Retrieval
	  • Overhead 8 KB per objek, dikenakan harga kelas penyimpanan `STANDARD`
	  Untuk informasi selengkapnya, lihat Transisi objek menggunakan Siklus Hidup Amazon S3.
	
	Untuk informasi selengkapnya tentang metrik penyimpanan, lihat Memantau metrik dengan Amazon CloudWatch.
		30 April 2018

Perubahan	Deskripsi	Tanggal
Kelas penyimpanan baru	Amazon S3 sekarang menawarkan kelas penyimpanan baru, `STANDARD_IA` (IA, untuk akses yang jarang) guna menyimpan objek. Kelas penyimpanan ini dioptimalkan untuk data yang tahan lama dan tidak banyak diakses. Untuk informasi selengkapnya, lihat Menggunakan kelas penyimpanan Amazon S3.	4 April 2018
Amazon S3 Select	Amazon S3 kini mendukung pengambilan konten objek berdasarkan ekspresi SQL. Untuk informasi selengkap nya, lihat Menyaring dan mengambil data menggunakan Amazon S3 Select.	4 April 2018
Wilayah Asia Pasifik (Osaka-Lokal)	Amazon S3 kini tersedia di Wilayah Asia Pasifik (Osaka-Lokal). Untuk informasi selengkapnya tentang Wilayah dan titik akhir Amazon S3, lihat Wilayah dan titik akhir di Referensi Umum AWS. **Important** Anda dapat menggunakan Wilayah Asia Pasifik (Osaka-Lokal) hanya dalam hubungannya dengan Wilayah Asia Pasifik (Tokyo). Untuk meminta akses ke Wilayah Asia Pasifik (Osaka-Lokal), hubungi perwakilan penjualan Anda.	12 Februari 2018
Stempel waktu pembuatan Inventaris Amazon S3	Inventaris Amazon S3 kini mencakup stempel waktu tanggal dan waktu mulai pembuatan laporan Inventaris Amazon S3. Anda dapat menggunakan stempel waktu untuk menentukan perubahan dalam penyimpanan Amazon S3 Anda sejak waktu mulai pembuatan laporan inventaris.	16 Januari 2018

Perubahan	Deskripsi	Tanggal
Wilayah Eropa (Paris)	Amazon S3 kini tersedia di Wilayah Eropa (Paris). Untuk informasi selengkapnya tentang Wilayah dan titik akhir Amazon S3, lihat Wilayah dan titik akhir di Referensi Umum AWS.	18 Desember 2017
Wilayah Tiongkok (Ningxia)	Amazon S3 kini tersedia di Wilayah Tiongkok (Ningxia). Untuk informasi selengkapnya tentang Wilayah dan titik akhir Amazon S3, lihat Wilayah dan titik akhir di Referensi Umum AWS.	29 November 2017
Dukungan untuk berkas Inventaris Amazon S3 yang diformat oleh ORC	Amazon S3 kini mendukung format kolom baris yang dioptimalkan (ORC) Apache selain format file nilai yang dipisahkan koma (CSV) untuk file output inventaris. Selain itu, Anda kini dapat melakukan kueri Inventaris Amazon S3 menggunakan SQL standar dengan menggunakan Amazon Athena, Amazon Redshift Spectrum, dan alat lainnya seperti Prasto, Apache Hive, dan Apache Spark. Untuk informasi selengkapnya, lihat Inventaris Amazon S3.	17 November 2017
Enkripsi default untuk bucket S3	Enkripsi default Amazon S3 menyediakan cara untuk menetapkan perilaku enkripsi default untuk bucket S3. Anda dapat menetapkan enkripsi default pada bucket sehingga semua objek dienkripsi saat mereka disimpan dalam bucket. Objek dienkripsi menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3) atau kunci terkelola (SSE-KMS). AWS Untuk informasi selengkapnya, lihat Mengatur perilaku enkripsi di sisi server default untuk bucket Amazon S3.	6 November 2017

Perubahan	Deskripsi	Tanggal
Status enkripsi dalam Inventaris Amazon S3	Amazon S3 sekarang mendukung penyertaan status enkripsi dalam Inventaris Amazon S3 sehingga Anda dapat melihat bagaimana objek Anda dienkripsi saat diam untuk audit kepatuhan atau tujuan lain. Anda juga dapat mengonfigurasi untuk mengenkripsi Inventaris Amazon S3 dengan enkripsi di sisi server (SSE) atau SSE-KMS sehingga semua file inventaris dienkripsi dengan semestiny a. Untuk informasi selengkapnya, lihat Inventaris Amazon S3.	6 November 2017
Peningkatan Replikasi Lintas-Wilayah (CRR)	Replikasi Lintas-Wilayah sekarang mendukung hal berikut: • Dalam skenario lintas akun, Anda dapat menambahkan konfigurasi CRR untuk mengubah kepemilikan replika ke Akun AWS yang menjadi pemilik bucket tujuan. Untuk informasi selengkapnya, lihat Mengubah pemilik replika. • Secara default, Amazon S3 tidak mereplikasi objek di bucket sumber Anda yang dibuat menggunakan enkripsi sisi server menggunakan kunci yang disimpan dalam AWS KMS Dalam konfigurasi CRR Anda, Anda sekarang dapat mengarahkan Amazon S3 untuk mereplikasi objek ini. Untuk informasi selengkapnya, lihat Mereplikasi objek terenkripsi (SSE-S3, SSE-KMS, SSE-KMS).	6 November 2017
Wilayah Eropa (London)	Amazon S3 kini tersedia di Wilayah Eropa (London). Untuk informasi selengkapnya tentang Wilayah dan titik akhir Amazon S3, lihat Wilayah dan titik akhir di Referensi Umum AWS.	13 Desember 2016
Wilayah Kanada (Pusat)	Amazon S3 kini tersedia di Wilayah Kanada (Pusat). Untuk informasi selengkapnya tentang Wilayah dan titik akhir Amazon S3, lihat Wilayah dan titik akhir di Referensi Umum AWS.	8 Desember 2016

Perubahan	Deskripsi	Tanggal
Penandaan objek	Amazon S3 kini mendukung penandaan objek. Penandaan objek memungkinkan Anda mengategorikan penyimpanan. Awalan nama kunci objek juga memungkinkan Anda untuk mengategorikan penyimpanan, penandaan objek menambah dimensi lain. Ada penawaran penandaan keuntungan tambahan. Ini termasuk: • Tag objek memungkinkan kontrol akses izin terperinci (misalnya, Anda dapat memberikan izin pengguna IAM ke objek hanya baca dengan tag tertentu). • Kontrol terperinci dalam menentukan konfigurasi siklus hidup. Anda dapat menetapkan tag untuk memilih subset objek yang menerapkan aturan siklus hidup. • Jika Replikasi Lintas-Wilayah (CRR) telah dikonfigurasi, Amazon S3 dapat mereplikasi tag. Anda harus memberikan izin yang diperlukan kepada peran IAM yang dibuat untuk Amazon S3 guna mengasumsikan objek yang direplikasi atas nama Anda. • Anda juga dapat menyesuaikan CloudWatch metrik dan CloudTrail peristiwa untuk menampilkan informasi berdasarkan filter tag tertentu. Untuk informasi selengkapnya, lihat Mengategorikan penyimpanan Anda menggunakan tag.	29 November 2016

Perubahan	Deskripsi	Tanggal
Siklus Hidup Amazon S3 kini mendukung filter berbasis tag	Amazon S3 kini mendukung pemfilteran berbasis tag dalam konfigurasi siklus hidup. Anda sekarang dapat menentukan aturan siklus hidup, yaitu Anda bisa menentukan awalan kunci, satu atau beberapa tag objek, atau kombinasi keduanya untuk memilih subset objek yang menerapkan aturan siklus hidup. Untuk informasi selengkapnya, lihat Mengelola siklus hidup penyimpanan Anda.	29 November 2016
CloudWatch minta metrik untuk bucket	Amazon S3 sekarang mendukung CloudWatch metrik untuk permintaan yang dibuat di bucket. Saat Anda mengaktifkan metrik ini untuk satu bucket, metrik melaporkan pada interval 1 menit. Anda juga dapat mengonfigurasi objek mana dalam bucket yang akan melaporkan metrik permintaan ini. Untuk informasi selengkapnya, lihat Memantau metrik dengan Amazon CloudWatch.	29 November 2016
Inventaris Amazon S3	Amazon S3 kini mendukung inventaris penyimpanan. Inventaris Amazon S3 menyediakan output berkas datar dari objek Anda dan metadata yang sesuai setiap hari atau setiap minggu untuk bucket S3 atau awalan bersama (yaitu, objek yang memiliki nama yang dimulai dengan string umum). Untuk informasi selengkapnya, lihat Inventaris Amazon S3.	29 November 2016

Perubahan	Deskripsi	Tanggal
Analitik Amazon S3–Analisis Kelas Penyimpanan	Analitik Amazon S3 yang baru–fitur analisis kelas penyimpanan mengamati pola akses data untuk membantu Anda menentukan kapan mengalihkan penyimpanan `STANDARD` yang jarang diakses ke kelas penyimpanan `STANDARD_IA` (IA, untuk akses yang jarang). Setelah analisis kelas penyimpanan mengamati pola akses yang jarang dari kumpulan data yang difilter selama periode waktu tertentu, Anda dapat menggunakan hasil analisis untuk membantu Anda meningkatkan konfigurasi siklus hidup. Fitur ini juga mencakup analisis terperinci harian mengenai penggunaan penyimpanan Anda di tingkat bucket, awalan, atau tag tertentu yang dapat Anda ekspor ke bucket S3.	29 November 2016
Pengambilan data baru yang Dipercepat dan Massal saat memulihkan objek yang diarsipkan dari S3 Glacier	Amazon S3 kini mendukung pengambilan data yang Dipercepat dan Massal di samping pengambilan Standar saat memulihkan objek yang diarsipkan ke S3 Glacier. Untuk informasi selengkapnya, lihat Memulihkan objek yang diarsipkan.	21 November 2016
CloudTrail pencatatan objek	CloudTrail mendukung pencatatan operasi API tingkat objek Amazon S3 seperti`GetObject` ,`PutObject` , dan. `DeleteObject` Anda dapat mengonfigurasi penyeleksi peristiwa Anda untuk log operasi API tingkat objek. Untuk informasi selengkapnya, lihat Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail.	21 November 2016
Wilayah AS Timur (Ohio)	Amazon S3 kini tersedia di Wilayah AS Timur (Ohio). Untuk informasi selengkapnya tentang Wilayah dan titik akhir Amazon S3, lihat Wilayah dan titik akhir di Referensi Umum AWS.	17 Oktober 2016

Perubahan	Deskripsi	Tanggal
Dukungan IPv6 untuk Amazon S3 Transfer Acceleration	Amazon S3 kini mendukung Internet Protocol versi 6 (IPv6) untuk Amazon S3 Transfer Acceleration. Anda dapat terhubung ke Amazon S3 melalui IPv6 dengan menggunak an tumpukan ganda untuk titik akhir Transfer Acceleration yang baru. Untuk informasi selengkapnya, lihat Memulai Amazon S3 Transfer Acceleration.	6 Oktober 2016
Dukungan IPv6	Amazon S3 kini mendukung Protokol Internet versi 6 (IPv6). Anda dapat mengakses Amazon S3 melalui IPv6 dengan menggunakan titik akhir tumpukan ganda. Untuk informasi selengkapnya, lihat Membuat permintaan ke Amazon S3 melalui IPv6.	11 Agustus, 2016
Wilayah Asia Pasifik (Mumbai)	Amazon S3 kini tersedia di Wilayah Asia Pasifik (Mumbai). Untuk informasi selengkapnya tentang Wilayah dan titik akhir Amazon S3, lihat Wilayah dan titik akhir di Referensi Umum AWS.	27 Juni 2016
Amazon S3 Transfer Acceleration	Amazon S3 Transfer Acceleration memungkinkan transfer berkas yang cepat, mudah, dan aman melalui jarak jauh antara klien Anda dan bucket S3. Transfer Acceleration memanfaatkan lokasi edge Amazon yang didistribusikan CloudFront secara global. Untuk informasi selengkapnya, lihat Mengonfigurasi transfer file yang cepat dan aman menggunakan Amazon S3 Transfer Acceleration.	19 April 2016
Dukungan siklus hidup untuk menghapus penanda hapus objek kedaluwarsa	Tindakan `Expiration` konfigurasi siklus hidup kini memungkinkan Anda mengarahkan Amazon S3 untuk menghapus penanda hapus objek yang kedaluwarsa di bucket berversi. Untuk informasi selengkapnya, lihat Elemen untuk mendeskripsikan tindakan siklus hidup.	16 Maret 2016

Perubahan	Deskripsi	Tanggal
Konfigurasi siklus hidup bucket kini mendukung tindakan untuk menghenti kan pengunggahan multibagian yang tidak lengkap	Konfigurasi siklus hidup bucket kini mendukung tindakan `AbortIncompleteMultipartUpload` yang dapat Anda gunakan untuk mengarahkan Amazon S3 untuk menghentikan pengunggahan multibagian yang tidak selesai dalam jumlah hari yang ditentukan setelah diinisias i. Saat pengunggahan multibagian memenuhi syarat untuk operasi penghentian, Amazon S3 menghapus setiap bagian yang diunggah dan menghentikan pengunggahan multibagian. Untuk informasi konseptual, lihat topik berikut di Panduan Pengguna Amazon S3: • Membatalkan unggahan multibagian • Elemen untuk mendeskripsikan tindakan siklus hidup Operasi API berikut telah diperbarui untuk mendukung tindakan baru: • PUT Bucket lifecycle–Konfigurasi XML sekarang memungkinkan Anda menentukan tindakan `AbortInco mpleteMultipartUpload` dalam aturan konfigura si siklus hidup. • List Parts dan Initiate Multipart Upload–Kedua operasi API ini sekarang mengembalikan dua header respons tambahan (`x-amz-abort-date` , dan `x-amz-abort-rule-id`) jika bucket memiliki aturan siklus hidup yang menentukan tindakan `AbortInco mpleteMultipartUpload` . Header dalam respons ini menunjukkan saat unggahan multibagian yang dimulai memenuhi syarat untuk operasi penghentian dan aturan siklus hidup mana yang berlaku.	16 Maret 2016

Perubahan	Deskripsi	Tanggal
Wilayah Asia Pasifik (Seoul)	Amazon S3 kini tersedia di Wilayah Asia Pasifik (Seoul). Untuk informasi selengkapnya tentang Wilayah dan titik akhir Amazon S3, lihat Wilayah dan titik akhir di Referensi Umum AWS.	6 Januari 2016
Kunci kondisi baru dan perubahan pengunggahan multibagian	Kebijakan IAM kini mendukung kunci kondisi `s3:x-amz-storage-class` Amazon S3. Untuk informasi selengkapnya, lihat Contoh kebijakan bucket menggunakan tombol kondisi. Anda tidak perlu lagi menjadi inisiator pengunggahan multibagian untuk mengunggah bagian dan menyelesaikan pengunggahan. Untuk informasi selengkapnya, lihat API dan izin unggahan multibagian.	14 Desember 2015
Mengganti nama Wilayah Standar AS	Mengubah string nama Wilayah dari "Standar AS” menjadi "AS Timur (Virginia Utara).” Ini hanya pembaruan nama Wilayah, tidak ada perubahan dalam fungsionalitas.	11 Desember 2015

Perubahan	Deskripsi	Tanggal
Kelas penyimpanan baru	Amazon S3 sekarang menawarkan kelas penyimpanan baru, STANDARD_IA (IA, untuk akses yang jarang) guna menyimpan objek. Kelas penyimpanan ini dioptimalkan untuk data yang tahan lama dan tidak banyak diakses. Untuk informasi selengkapnya, lihat Menggunakan kelas penyimpanan Amazon S3. Pembaruan fitur konfigurasi siklus hidup sekarang memungkinkan Anda melakukan transisi objek ke kelas penyimpanan STANDARD_IA. Untuk informasi selengkap nya, lihat Mengelola siklus hidup penyimpanan Anda. Sebelumnya, fitur Replikasi Lintas-Wilayah menggunak an kelas penyimpanan objek sumber untuk replika objek. Sekarang, saat Anda mengonfigurasi Replikasi Lintas-Wilayah, Anda dapat menentukan kelas penyimpanan untuk replika objek yang dibuat dalam bucket tujuan. Untuk informasi selengkapnya, lihat Mereplikasi ikhtisar objek.	16 September 2015
AWS CloudTrail integrasi	AWS CloudTrail Integrasi baru memungkinkan Anda merekam aktivitas API Amazon S3 di bucket S3 Anda. Anda dapat menggunakan CloudTrail untuk melacak pembuatan atau penghapusan bucket S3, modifikasi kontrol akses, atau perubahan konfigurasi siklus hidup. Untuk informasi selengkapnya, lihat Mencatat panggilan API Amazon S3 menggunakan AWS CloudTrail.	1 September 2015

Perubahan	Deskripsi	Tanggal
Peningkatan batas bucket	Amazon S3 kini mendukung peningkatan batas bucket. Secara default, pelanggan dapat membuat hingga 100 ember di dalamnya. Akun AWS Pelanggan yang memerlukan bucket tambahan dapat meningkatkan batas tersebut dengan mengajukan peningkatan batas layanan. Untuk informasi tentang cara meningkatkan batas bucket Anda, buka Layanan AWS kuota di AWS Referensi Umum. Untuk informasi lebih lanjut, lihat Menggunakan AWS SDK dan Pembatasan dan batasan bucket.	4 Agustus 2015
Pembaruan model konsistensi	Amazon S3 sekarang mendukung read-after-write konsistensi untuk objek baru yang ditambahkan ke Amazon S3 di Wilayah AS Timur (Virginia N.). Sebelum pembaruan ini, semua Wilayah kecuali Wilayah AS Timur (Virginia N.) mendukung read-after-write konsistensi untuk objek baru yang diunggah ke Amazon S3. Dengan peningkat an ini, Amazon S3 sekarang read-after-write mendukung konsistensi di semua Wilayah untuk objek baru yang ditambahkan ke Amazon S3. ead-after-writeKonsistensi R memungkinkan Anda untuk mengambil objek segera setelah pembuatan di Amazon S3. Untuk informasi selengkapnya, lihat Wilayah.	4 Agustus 2015
Pemberitahuan peristiwa	Notifikasi Peristiwa Amazon S3 telah diperbarui untuk menambahkan pemberitahuan saat objek dihapus dan untuk menambahkan pemfilteran pada nama objek dengan awalan dan suffix yang cocok. Untuk informasi selengkap nya, lihat Notifikasi Peristiwa Amazon S3.	28 Juli 2015

Perubahan	Deskripsi	Tanggal
CloudWatch Integrasi Amazon	CloudWatch Integrasi Amazon baru memungkinkan Anda memantau dan mengatur alarm pada penggunaan Amazon S3 Anda CloudWatch melalui metrik untuk Amazon S3. Metrik yang didukung meliputi total byte penyimpanan Standar, byte total untuk Reduced-Redundancy Storage, dan jumlah total objek untuk bucket S3 tertentu. Untuk informasi selengkapnya, lihat Memantau metrik dengan Amazon CloudWatch.	28 Juli 2015
Dukungan untuk menghapus dan mengosongkan bucket berisi	Amazon S3 kini mendukung penghapusan dan pengosong an bucket berisi. Untuk informasi selengkapnya, lihat Mengosongkan bucket.	16 Juli 2015
Kebijakan bucket untuk titik akhir VPC Amazon	Amazon S3 telah menambahkan dukungan kebijakan bucket untuk titik akhir cloud privat virtual (VPC). Anda dapat menggunakan kebijakan bucket S3 untuk mengontro l akses ke bucket dari titik akhir VPC tertentu, atau VPC tertentu. Titik akhir VPC mudah dikonfigurasi, sangat andal, dan menyediakan koneksi aman ke Amazon S3 tanpa memerlukan gateway atau instans NAT. Untuk informasi selengkapnya, lihat Megendalikan akses dari titik akhir VPC dengan kebijakan bucket.	29 April 2015
Pemberitahuan peristiwa	Pemberitahuan Acara Amazon S3 telah diperbarui untuk mendukung peralihan ke izin berbasis sumber daya untuk fungsi. AWS Lambda Untuk informasi selengkapnya, lihat Notifikasi Peristiwa Amazon S3.	9 April 2015
Replikasi Lintas-Wi layah	Amazon S3 kini mendukung Replikasi Lintas-Wilayah. Replikasi Lintas Wilayah adalah penyalinan objek secara otomatis dan asinkron di seluruh ember dengan cara yang berbeda. Wilayah AWS Untuk informasi selengkapnya, lihat Mereplikasi ikhtisar objek.	24 Maret 2015

Perubahan	Deskripsi	Tanggal
Pemberitahuan peristiwa	Amazon S3 kini mendukung tipe dan tujuan peristiwa baru dalam konfigurasi pemberitahuan bucket. Sebelum rilis ini, Amazon S3 hanya mendukung s3: jenis ReducedRe dundancyLostObject acara dan topik Amazon SNS sebagai tujuan. Untuk informasi selengkapnya tentang tipe peristiwa baru, lihat Notifikasi Peristiwa Amazon S3.	13 November 2014
Enkripsi sisi server dengan kunci enkripsi yang disediakan pelanggan	Enkripsi sisi server dengan kunci AWS Key Management Service (AWS KMS) (SSE-KMS) Amazon S3 sekarang mendukung enkripsi sisi server menggunakan. AWS KMS Fitur ini memungkinkan Anda mengelola kunci amplop AWS KMS, dan AWS KMS panggilan Amazon S3 untuk mengakses kunci amplop dalam izin yang Anda tetapkan. Untuk informasi selengkapnya tentang enkripsi sisi server dengan AWS KMS, lihat Melindungi Data Menggunak an Enkripsi Sisi Server dengan. AWS Key Management Service	12 November 2014
Wilayah Eropa (Frankfurt)	Amazon S3 kini tersedia di Wilayah Eropa (Frankfurt).	23 Oktober 2014

Perubahan	Deskripsi	Tanggal
Enkripsi di sisi server dengan kunci enkripsi yang disediakan pelanggan	Amazon S3 kini mendukung enkripsi di sisi server menggunakan kunci enkripsi yang disediakan pelanggan (SSE-C). Enkripsi di sisi server memungkinkan Anda meminta Amazon S3 mengenkripsi data diam Anda. Saat menggunakan SSE-C, Amazon S3 mengenkripsi objek Anda dengan kunci enkripsi khusus yang Anda sediakan. Karena Amazon S3 melakukan enkripsi untuk Anda, Anda mendapatkan manfaat menggunakan kunci enkripsi Anda sendiri tanpa perlu biaya penulisan atau mengeksekusi kode enkripsi Anda sendiri. Untuk informasi selengkapnya tentang SSE-C, lihat enkripsi di sisi server (Menggunakan Kunci Enkripsi yang Disediakan Pelanggan).	12 Juni 2014
Dukungan siklus hidup untuk Penentuan Versi	Sebelum rilis ini, konfigurasi siklus hidup hanya didukung pada bucket tanpa versi. Sekarang Anda dapat mengonfig urasi siklus hidup di bucket tanpa versi dan dengan dukungan Penentuan Versi. Untuk informasi selengkapnya, lihat Mengelola siklus hidup penyimpanan Anda.	20 Mei 2014
Topik kontrol akses direvisi	Revisi dokumentasi kontrol akses Amazon S3. Untuk informasi selengkapnya, lihat Identity and Access Management untuk Amazon S3.	15 April 2014
Topik pencatatan log akses server direvisi	Revisi dokumentasi pencatatan akses server. Untuk informasi selengkapnya, lihat Pencatatan permintaan dengan pencatatan akses server.	26 November 2013
Sampel .NET SDK diperbarui ke versi 2.0	Sampel .NET SDK dalam panduan ini sekarang sesuai dengan versi 2.0.	26 November 2013

Perubahan	Deskripsi	Tanggal
Dukungan SOAP Melalui HTTP Tidak Berlaku Lagi	Dukungan SOAP melalui HTTP dihilangkan, tetapi masih tersedia melalui HTTPS. Fitur Amazon S3 baru tidak mendukung SOAP. Kami menyarankan Anda menggunakan REST API atau AWS SDK.	20 September 2013
Dukungan variabel kebijakan IAM	Bahasa kebijakan IAM kini mendukung variabel. Ketika suatu kebijakan dievaluasi, setiap variabel kebijakan diganti dengan nilai yang dipasok oleh informasi berbasis konteks dari sesi pengguna yang diautentikasi. Anda dapat menggunakan variabel kebijakan untuk menentukan kebijakan tujuan umum tanpa mencantumkan semua komponen kebijakan secara eksplisit. Untuk informasi selengkapnya tentang variabel kebijakan, lihat Ikhtisar Variabel Kebijakan IAM dalam Panduan Pengguna IAM. Untuk contoh variabel kebijakan di Amazon S3, lihat Contoh kebijakan berbasis identitas untuk Amazon S3.	3 April 2013
Dukungan konsol untuk Pembayaran oleh Pemohon	Sekarang Anda dapat mengonfigurasi bucket untuk Pembayaran oleh Pemohon dengan menggunakan konsol Amazon S3. Untuk informasi selengkapnya, lihat Menggunakan bucket Pembayaran Pemohon untuk transfer dan penggunaan penyimpanan.	31 Desember 2012

Perubahan	Deskripsi	Tanggal
Dukungan domain root untuk hosting situs web	Amazon S3 kini mendukung hosting situs web statis pada domain root. Pengunjung situs web Anda dapat mengakses situs Anda dari peramban mereka tanpa menyebutkan www di alamat web (misalnya, mereka dapat menggunakan example.com, alih-alih www.example.com). Banyak pelanggan sudah melakukan hosting situs web statis di Amazon S3 yang dapat diakses dari subdomain www (misalnya, www.example.com). Sebelumnya, untuk mendukung akses domain root, Anda perlu menjalank an server web Anda sendiri untuk memproksi permintaa n domain root dari peramban ke situs web Anda di Amazon S3. Menjalankan server web ke permintaan proksi menimbulkan biaya tambahan, beban operasion al, dan potensi titik kegagalan lainnya. Sekarang, Anda dapat memanfaatkan ketersediaan dan daya tahan tinggi Amazon S3 untuk alamat domain www dan root. Untuk informasi selengkapnya, lihat Hosting situs web statis menggunakan Amazon S3.	27 Desember 2012
Revisi konsol	Konsol Amazon S3 telah diperbarui. Topik dokumentasi yang mengacu pada konsol telah direvisi sebagaimana mestinya.	14 Desember 2012
Dukungan untuk Mengarsipkan Data ke S3 Glacier	Amazon S3 kini mendukung opsi penyimpanan yang memungkinkan Anda memanfaatkan layanan penyimpanan berbiaya rendah S3 Glacier untuk pengarsipan data. Untuk mengarsipkan objek, Anda menentukan aturan pengarsip an yang mengidentifikasi objek dan kerangka waktu saat Anda ingin Amazon S3 mengarsipkan objek tersebut ke S3 Glacier. Anda dapat dengan mudah menetapkan aturan pada bucket menggunakan konsol Amazon S3 atau secara terprogram menggunakan Amazon S3 API atau SDK. AWS Untuk informasi selengkapnya, lihat Mengelola siklus hidup penyimpanan Anda.	13 November 2012

Perubahan	Deskripsi	Tanggal
Dukungan untuk Pengalihan Halaman Situs Web	Untuk bucket yang dikonfigurasi sebagai situs web, Amazon S3 kini mendukung pengalihan permintaan objek ke objek lain dalam bucket yang sama atau ke URL eksternal. Untuk informasi selengkapnya, lihat (Opsional) Mengonfigurasi pengalihan halaman web. Untuk informasi tentang hosting situs web, lihat Hosting situs web statis menggunakan Amazon S3.	4 Oktober 2012
Dukungan untuk Berbagi Sumber Daya Lintas Asal (CORS)	Amazon S3 kini mendukung Berbagi Sumber Daya Lintas Asal (CORS). CORS mendefinisikan cara aplikasi web klien yang dimuat dalam satu domain yang dapat berintera ksi dengan atau mengakses sumber daya dalam domain yang berbeda. Dengan dukungan CORS di Amazon S3, Anda dapat membangun aplikasi web sisi klien yang kaya di atas Amazon S3 dan secara selektif memungkinkan akses lintas domain ke sumber daya Amazon S3 Anda. Untuk informasi selengkapnya, lihat Berbagi sumber daya lintas asal (CORS).	31 Agustus 2012
Dukungan untuk Tanda Alokasi Biaya	Amazon S3 kini mendukung penandaan alokasi biaya, yang memungkinkan Anda memberi label pada bucket S3 sehingga Anda dapat melacak biayanya terhadap proyek atau kriteria lainnya dengan lebih mudah. Untuk informasi selengkapnya tentang penandaan untuk bucket, lihat Menggunakan tag alokasi biaya bucket S3.	21 Agustus 2012

Perubahan	Deskripsi	Tanggal
Dukungan untuk akses API yang dilindungi MFA dalam kebijakan bucket	Amazon S3 sekarang mendukung akses API yang dilindungi MFA, fitur yang dapat menerapkan Otentikasi AWS Multi-Faktor untuk tingkat keamanan ekstra saat mengakses sumber daya Amazon S3 Anda. Ini adalah fitur keamanan yang mengharuskan pengguna untuk membuktikan kepemilikan fisik perangkat MFA dengan menyediakan kode MFA yang valid. Untuk informasi selengkapnya, kunjungi Autentikasi Multi-Faktor AWS. Sekarang Anda dapat meminta autentikasi MFA untuk permintaan akses sumber daya Amazon S3. Untuk menerapkan autentikasi MFA, Amazon S3 kini mendukung kunci `aws:MultiFactorAuthAge` dalam kebijakan bucket. Untuk contoh kebijakan bucket, lihat Membutuhkan MFA.	10 Juli 2012
Dukungan Kedaluwarsa Objek	Anda dapat menggunakan Kedaluwarsa Objek untuk menjadwalkan penghapusan data otomatis setelah periode waktu yang dikonfigurasi. Anda mengatur kedaluwarsa objek dengan menambahkan konfigurasi siklus hidup ke bucket.	27 Desember 2011
Wilayah baru yang didukung	Amazon S3 kini mendukung Wilayah Amerika Selatan (Sao Paulo). Untuk informasi selengkapnya, lihat Mengakses dan mendaftarkan bucket Amazon S3.	14 Desember 2011
Penghapusan Multi-Objek	Amazon S3 kini mendukung API Penghapusan Multi-Objek yang memungkinkan Anda menghapus beberapa objek dalam satu permintaan. Dengan fitur ini, Anda dapat menghapus banyak objek dari Amazon S3 lebih cepat dibandingkan dengan menggunakan beberapa permintaan DELETE individual. Untuk informasi selengkapnya, lihat Menghapus objek Amazon S3.	7 Desember 2011

Perubahan	Deskripsi	Tanggal
Wilayah baru yang didukung	Amazon S3 kini mendukung Wilayah AS Barat (Oregon). Untuk informasi selengkapnya, lihat Bucket dan Wilayah.	8 November 2011
Pembaruan Dokumentasi	Perbaikan bug dokumentasi.	8 November 2011
Pembaruan Dokumentasi	Selain perbaikan bug dokumentasi, rilis ini mencakup beberapa peningkatan berikut: • Bagian enkripsi sisi server baru menggunakan AWS SDK for PHP dan AWS SDK for Ruby (lihat). Menentukan enkripsi di sisi server dengan kunci terkelola Amazon S3 (SSE-S3)	17 Oktober 2011
Dukungan enkripsi di sisi server	Amazon S3 kini mendukung enkripsi di sisi server. Dengan ini, Anda dapat meminta Amazon S3 untuk mengenkripsi data diam, yaitu mengenkripsi data objek Anda ketika Amazon S3 menulis data Anda ke disk di pusat datanya. Selain pembaruan REST API, AWS SDK for Java dan .NET menyediakan fungsionalitas yang diperlukan untuk meminta enkripsi sisi server. Anda juga dapat meminta enkripsi di sisi server saat mengunggah objek menggunakan file AWS Management Console. Untuk mempelajari enkripsi data lebih lanjut, kunjungi Menggunakan Enkripsi Data.	4 Oktober 2011

Perubahan	Deskripsi	Tanggal
Pembaruan Dokumentasi	Selain perbaikan bug dokumentasi, rilis ini mencakup beberapa peningkatan berikut: • Menambahkan sampel Ruby dan PHP ke bagian Membuat permintaan. • Menambahkan bagian yang menjelaskan cara membuat dan menggunakan URLs yang telah ditandatangani sebelumnya. Untuk informasi lebih lanjut, lihat Berbagi objek dengan URL yang telah ditandatangani dan Berbagi objek dengan URL yang telah ditandatangani. • Memperbarui bagian yang ada untuk memperkenalkan AWS Explorers for Eclipse dan Visual Studio. Untuk informasi selengkapnya, lihat Mengembangkan dengan Amazon S3 menggunakan SDK AWS.	22 September 2011

Perubahan	Deskripsi	Tanggal
Dukungan untuk mengirim permintaan menggunakan kredensial keamanan sementara	Selain menggunakan kredensil keamanan pengguna Anda Akun AWS dan IAM untuk mengirim permintaan yang diautentikasi ke Amazon S3, Anda sekarang dapat mengirim permintaan menggunakan kredensil keamanan sementara yang Anda peroleh dari (IAM). AWS Identity and Access Management Anda dapat menggunakan AWS Security Token Service API atau pustaka pembungkus AWS SDK untuk meminta kredensyal sementara ini dari IAM. Anda dapat meminta kredensial keamanan sementara ini untuk penggunaan Anda sendiri atau menyerahkannya kepada pengguna gabungan dan aplikasi. Fitur ini memungkinkan Anda untuk mengelola pengguna Anda di luar AWS dan memberi mereka kredensyal keamanan sementara untuk mengakses sumber daya Anda AWS . Untuk informasi selengkapnya, lihat Membuat permintaan. Untuk informasi selengkapnya tentang dukungan IAM untuk kredensial keamanan sementara, lihat Kredensial Keamanan Sementara dalam Panduan Pengguna IAM.	3 Agustus 2011
API Unggahan Multibagian diperluas untuk memungkinkan penyalinan objek hingga 5 TB	Sebelum rilis ini, API Amazon S3 mendukung penyalinan objek dengan ukuran hingga 5 GB. Untuk mengaktifkan penyalinan objek yang lebih besar dari 5 GB, Amazon S3 kini memperluas API unggahan multibagian dengan operasi baru, `Upload Part (Copy)` . Anda dapat menggunakan operasi pengunggahan multibagian ini untuk menyalin objek berukuran hingga 5 TB. Untuk informasi selengkapnya, lihat Menyalin, memindahkan, dan mengganti nama objek. Untuk informasi konseptual tentang API unggahan multibagian, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.	21 Juni 2011

Perubahan	Deskripsi	Tanggal
Panggilan API SOAP melalui HTTP dinonaktifkan	Untuk meningkatkan keamanan, panggilan API SOAP melalui HTTP dinonaktifkan. Permintaan SOAP yang diautentikasi dan anonim harus dikirim ke Amazon S3 menggunakan SSL.	6 Juni 2011
IAM memungkinkan delegasi lintas akun	Sebelumnya, untuk mengakses sumber daya Amazon S3, pengguna IAM memerlukan izin dari induk dan pemilik sumber daya Akun AWS Amazon S3. Dengan akses lintas akun, pengguna IAM sekarang hanya perlu izin dari akun pemilik. Artinya, Jika pemilik sumber daya memberikan akses ke Akun AWS, sekarang Akun AWS dapat memberikan pengguna IAM-nya akses ke sumber daya ini. Untuk informasi selengkapnya, lihat Membuat peran untuk mendelegasikan izin ke pengguna IAM dalam Panduan pengguna IAM. Untuk informasi selengkapnya tentang menentukan pengguna utama dalam kebijakan bucket, lihat Prinsipal untuk kebijakan bucket.	6 Juni 2011
Tautan baru	Informasi titik akhir layanan ini sekarang terletak di Referensi Umum AWS . Untuk informasi selengkapnya, buka Wilayah dan titik akhir dalam Referensi Umum AWS.	1 Maret 2011

Perubahan	Deskripsi	Tanggal
Dukungan untuk hosting situs web statis di Amazon S3	Amazon S3 memperkenalkan dukungan yang ditingkat kan untuk hosting situs web statis. Ini termasuk dukungan untuk dokumen indeks dan dokumen kesalahan kustom. Saat menggunakan fitur ini, permintaan ke root bucket atau subfolder (misalnya, `http://mywebsite.com/subfolder`) akan menampilkan dokumen indeks, alih-alih daftar objek di bucket Anda. Jika terjadi kesalahan, Amazon S3 akan menampilkan pesan kesalahan khusus Anda, alih-alih pesan kesalahan Amazon S3. Untuk informasi selengkapnya, lihat Hosting situs web statis menggunakan Amazon S3.	6 Juni 2011
Informasi titik akhir layanan ini sekarang terletak di Referensi Umum AWS . Untuk informasi selengkap nya, buka Wilayah dan titik akhir dalam Referensi Umum AWS.	1 Maret 2011	
Dukungan untuk hosting situs web statis di Amazon S3	Amazon S3 memperkenalkan dukungan yang ditingkat kan untuk hosting situs web statis. Ini termasuk dukungan untuk dokumen indeks dan dokumen kesalahan kustom. Saat menggunakan fitur ini, permintaan ke root bucket atau subfolder (misalnya, `http://mywebsite.com/subfolder`) akan menampilkan dokumen indeks, alih-alih daftar objek di bucket Anda. Jika terjadi kesalahan, Amazon S3 akan menampilkan pesan kesalahan khusus Anda, alih-alih pesan kesalahan Amazon S3. Untuk informasi selengkapnya, lihat Hosting situs web statis menggunakan Amazon S3.	17 Februari 2011

Perubahan	Deskripsi	Tanggal
Dukungan API Header Respons	API REST GET Object kini memungkinkan Anda mengubah header respons permintaan REST GET Object untuk setiap permintaan. Artinya, Anda dapat mengubah metadata objek dalam respons, tanpa mengubah objek itu sendiri. Untuk informasi selengkapnya, lihat Mengunggah objek.	14 Januari 2011
Dukungan objek besar	Amazon S3 telah meningkatkan ukuran maksimum objek yang dapat Anda simpan dalam bucket S3 dari 5 GB menjadi 5 TB. Jika Anda menggunakan API REST, Anda dapat mengunggah objek hingga 5 GB dalam satu operasi PUT. Untuk objek yang lebih besar, Anda harus menggunakan API REST Pengunggahan Multibagian untuk mengunggah objek secara terpisah. Untuk informasi selengkapnya, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.	9 Desember 2010
Pengunggahan multibagian	Pengunggahan multibagian memungkinkan unggahan yang lebih cepat dan lebih fleksibel ke Amazon S3. Dengan ini, Anda dapat mengunggah satu objek sebagai sekumpulan bagian. Untuk informasi selengkapnya, lihat Mengunggah dan menyalin objek menggunakan unggahan multibagian.	10 November 2010
Dukungan ID kanonis dalam kebijakan bucket	Anda kini dapat menentukan ID kanonis dalam kebijakan bucket. Untuk informasi selengkapnya, lihat Prinsipal untuk kebijakan bucket	17 September 2010
Amazon S3 bekerja dengan IAM	Layanan ini sekarang terintegrasi dengan AWS Identity and Access Management (IAM). Untuk informasi selengkap nya, kunjungi Layanan AWS yang bekerja dengan IAM di Panduan Pengguna IAM.	2 September 2010

Perubahan	Deskripsi	Tanggal
Pemberitahuan	Fitur pemberitahuan Amazon S3 memungkinkan Anda untuk mengonfigurasikan bucket sehingga Amazon S3 menerbitkan pesan ke topik Amazon Simple Notification Service (Amazon SNS) ketika Amazon S3 mendeteksi peristiwa penting pada bucket. Untuk informasi selengkap nya, lihat Mengatur Pemberitahuan Peristiwa Bucket.	14 Juli 2010
Kebijakan bucket	Kebijakan bucket adalah sistem manajemen akses yang Anda gunakan untuk mengatur izin akses di seluruh bucket, objek, dan kumpulan objek. Fungsi ini melengkap i dan dalam banyak kasus menggantikan daftar kontrol akses. Untuk informasi selengkapnya, lihat Kebijakan bucket untuk Amazon S3.	6 Juli 2010
Sintaksis gaya jalur tersedia di semua Wilayah	Amazon S3 kini mendukung sintaksis gaya jalur untuk bucket mana pun di Wilayah Klasik AS, atau jika bucket berada di Wilayah yang sama dengan titik akhir permintaa n. Untuk informasi selengkapnya, lihat Hosting Virtual.	9 Juni 2010
Titik akhir baru untuk Eropa (Irlandia)	Amazon S3 kini menyediakan titik akhir untuk Eropa (Irlandia): `http://s3-eu-west-1.amazonaws.com` .	9 Juni 2010
Konsol	Anda kini dapat menggunakan Amazon S3 melalui AWS Management Console. Anda dapat membaca tentang semua fungsionalitas Amazon S3 di konsol dalam Panduan Pengguna Amazon Simple Storage Service.	9 Juni 2010
Pengurangan Redundansi	Amazon S3 kini memungkinkan Anda mengurangi biaya penyimpanan dengan menyimpan objek di Amazon S3 dengan pengurangan redundansi. Untuk informasi selengkapnya, lihat Reduced Redundancy Storage.	12 Mei 2010
Wilayah baru yang didukung	Amazon S3 kini mendukung Wilayah Asia Pasifik (Singapur a). Untuk informasi selengkapnya, lihat Bucket dan Wilayah.	28 April 2010

Perubahan	Deskripsi	Tanggal
Penentuan Versi Objek	Rilis ini memperkenalkan Penentuan Versi objek. Semua objek sekarang dapat memiliki kunci dan versi. Jika Anda mengaktifkan Penentuan Versi untuk bucket, Amazon S3 memberikan ID versi unik kepada semua objek yang ditambahkan ke bucket. Fitur ini memungkinkan Anda memulihkan dari penimpaan dan penghapusan yang tidak diinginkan. Untuk informasi selengkapnya, lihat Penentuan Versi dan Menggunakan Penentuan Versi.	8 Februari 2010
Wilayah baru yang didukung	Amazon S3 kini mendukung Wilayah AS Barat (Californ ia Utara). Titik akhir baru untuk permintaan pada Wilayah ini adalah `s3-us-west-1.amazonaws.com` . Untuk informasi selengkapnya, lihat Bucket dan Wilayah.	2 Desember 2009
AWS SDK for .NET	AWS sekarang menyediakan pustaka, kode sampel, tutorial, dan sumber daya lainnya untuk pengembang perangkat lunak yang lebih suka membangun aplikasi menggunakan operasi API khusus bahasa.NET daripada REST atau SOAP. Pustaka ini menyediakan fungsi dasar (tidak disertakan dalam API REST atau SOAP), seperti autentikasi permintaan, percobaan ulang permintaan, dan penanganan kesalahan sehingga lebih mudah untuk memulai. Untuk informasi selegnkapnya tentang pustaka dan sumber daya khusus bahasa, lihat Mengembangkan dengan Amazon S3 menggunakan SDK AWS.	11 November 2009

AWSGlosarium

Untuk AWS terminologi terbaru, lihat AWSglosarium di Referensi. Glosarium AWS

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.